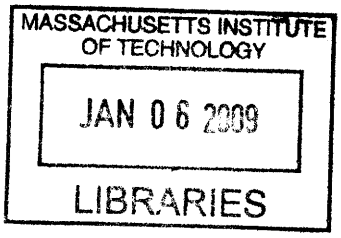
# Virtual Private Milieus:

*Sharing Our Digital Aura through Social and Physical Proximity*

Nadav Aharony

B.Sc. in Electrical Engineering, Technion – Israel Institute of Technology, July 2004

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning,
in partial fulfillment of the requirements for the degree of
Master of Science in Media Arts and Sciences
at the Massachusetts Institute of Technology




September 2008



Author

Nadav Aharony
Program in Media Arts and Sciences
August 8, 2008

Certified by

Andrew B. Lippman
Senior Research Scientist
MIT Media Laboratory

Accepted by

Deb Roy
Chair, Academic Program in Media Arts and Sciences
Program in Media Arts and Sciences

# Virtual Private Milieus:

*Sharing Our Digital Aura through Social and Physical Proximity*

Nadav Aharony

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning on August 8, 2008
in partial fulfillment of the requirements for the degree of
Master of Science in Media Arts and Sciences
at the Massachusetts Institute of Technology

## Abstract

People are quite good at establishing a social style and using it in different communications contexts, but they do less well when the communication is mediated by computer networks. It is hard to control what information is revealed and how one's digital persona will be presented or interpreted. In this thesis, we ameliorate this problem by creating a "*Virtual Private Milieu*", a "VPM", that allows networked devices to act on our behalf and project a "digital aura" to other people and devices around us in a manner analogous to the way humans naturally interact with one another. The dynamic aggregation of the different auras and facets that the devices expose to one another creates social spheres of interaction between sets of active devices, and consequently between people. We focus on the subset of networking that deals with proximate communication, which we dub *Face-to-Face Networking (FtFN).* Network interaction in this space is often analogous to human face-to-face interaction, and increasingly, our devices are being used in local situations. We describe a VPM framework, key features of which include the incorporation of trust and context parameters into the discovery and communication process, incorporation of multiple context-unique identities, and also the support for multiple degrees of security and privacy. We also present the "*Social Dashboard*", a readily usable control for one's aura. Finally, we review "*Comm.unity*", a software package that allows developers and researchers easy implementation and deployment of local and distant social applications, and present two applications developed over this platform.

Thesis Supervisor
Andrew P. Lippman
Senior Research Scientist
MIT Media Laboratory

# Virtual Private Milieus:

*Sharing Our Digital Aura through Social and Physical Proximity*

Nadav Aharony

The following people served as readers for this thesis:

Thesis Reader

David P. Reed
Adjunct Professor of Media Arts and Sciences
MIT Media Laboratory

Thesis Reader

Alex (Sandy) Pentland
Toshiba Professor of Media Arts and Sciences
MIT Media Laboratory

Thesis Reader

Judith Donath
Associate Professor of Media Arts and Sciences
MIT Media Laboratory

# Contents

# List of Figures

# Acknowledgements

There are many who provided me with invaluable support towards the completion of this thesis, during the last two years at MIT, and in general, as part of my life.

I would like to thank:

My advisor Andy Lippman, for the freedom he gave me to explore different research directions in the search of my passion, for being patient with me as I found ten different passions, and especially for always asking the right questions at the right time.

David P. Reed, my co-advisor, for believing in my ideas and always finding new ways and angles to look at them.

My readers, Alex (Sandy) Pentland and Judith Donath, for their support and their inspiration that helped me formulate my research direction.

My friends and thesis buddies, who influenced this work as it was evolving – Jamie Zigelbaum, Marcelo Cohelo, Anita Lillie, Alyssa Wright and Aninna Rüst – the k-up bunch, and also Mikey Siegel.

The numerous faculty members, teachers, colleagues, collaborators and friends with whom I have talked about this work in some form or other and helped shape it, if only by providing a listening ear: Mike Bove, Chris Csikszentmihályi, David Gautier, Neil Gershenfeld, Henry Holtzman, Kwan Hong Lee, Ellen Hume, Fulu Li, Anmol Madan, Pattie Maes, David Merril, Marvin Minsky, Manas Mittal , Bo Morgan, Frank Moss, Daniel Olguin Olguin, Mitch Resnick, Whitman Richards, Ron Rivest, Jhonatan Rotberg, Sajid Sadi, Yaar Schnitman, Dawei Shen, Dustin Smith, Dimitris Vyzovitis, Ben Waber, Grace Woo, and Pol Ypodimatopoulos.

The kindness and support of the lab staff - Deb Widener, Sandy Sener, Paula Aguilera, Jon Ferguson and the entire Necsys team, and special thanks to Linda Peterson and Gigi Shafer, without whom this thesis would not have seen the light of day.

MIT, the Media Laboratory, Viral Communications research group, MIT Center for Future Civic Media, Nokia and Paul Wisner in particular, the Knight Foundation, and 1369 Coffee House, where most of this document was written. I never drank so much coffee in my life.

Most importantly, I want to thank my family– my parents, my sister, and Maya most of all. You lent your shoulders. You are my giants.

*This thesis is dedicated to Maya.*

*I could not have done this without your support.*

*I could not have done this without you.*

*אהובה*

# Personal Note

*"Cavalcare la Tigre"*
*(Ride the Tiger)*

*-Julius Evola, 1961*

The Media Lab being what it is, and me being part of what is It, made it quite hard to transition from sprinting from project to project, from demonstration to demonstration, and from one deadline to another, into a state of writing down and documenting the concepts and ideas that have been bouncing inside my head throughout the past year. For nearly three months I stared at my reflection staring back at me from a blank page on my dusty computer screen. Slowly, some words began to trickle. Hesitant at first, they started gaining momentum. Then it happened. That moment, where, like the instant when a jet crosses the barrier of sound, some invisible boundary was crossed. The written characters, with my thoughts alongside, started to gush out faster than my hands could type, and I did not dare to stop.

During the past year, my advisors, thesis readers, friends, colleagues, and family have seen me jump between projects, disciplines, theory, and practice. I deemed it important to use this opportunity to show (or prove?) that all were parts of the same. Since above all - when throwing out everything else - this work is about identities, components of self, and their presentation.

It took me some years to realize that Cubism substitutes all facets of an object simultaneously. As I was writing this thesis it dawned on me that this is probably the bedlam that I projected towards my thesis committee, and especially my advisors, through the past year. Although I wish it were possible to take in the picture all at once, the current medium forces it to be flattened out into a linear narrative. This is my attempt at weaving the parts into a logical plot, although this is but one of many storylines, one of many trails one could take through this rugged landscape. Luckily, the medium is not completely flat.
Thank god for hyperlinks and cross-references.

This work attempts to bind together concepts from social and behavioral psychology, computer networks, sociology, artificial intelligence, comparative media, computer security, and new media studies. I truly cannot imagine being able to do it anywhere else but the MIT Media Laboratory. As I near the completion of this document, part of me muses over whether these words could only exist within the walls of the lab. The rest of me contemplates whether this work will survive if it is ever to be taken out of the sanctuary that is Building E15.

And with that, but one more jabber remains to be gibbered:

*"Once more unto the breach, dear friends, once more"*

*-William Shakespeare, Henry V, Act III, 1591*

*It was six men of Indostan*
*To learning much inclined,*
*Who went to see the Elephant*
*(Though all of them were blind),*
*That each by observation*
*Might satisfy his mind*

*The First approached the Elephant,*
*And happening to fall*
*Against his broad and sturdy side,*
*At once began to bawl:*
*"God bless me! but the Elephant*
*Is very like a wall!"*

*The Second, feeling of the tusk,*
*Cried, "Ho! what have we here*
*So very round and smooth and sharp?*
*To me 'tis mighty clear*
*This wonder of an Elephant*
*Is very like a spear!"*

*The Third approached the animal,*
*And happening to take*
*The squirming trunk within his hands,*
*Thus boldly up and spake:*
*"I see," quoth he, "the Elephant*
*Is very like a snake!"*

*The Fourth reached out an eager hand,*
*And felt about the knee.*
*"What most this wondrous beast is like*
*Is mighty plain," quoth he;*
*" 'Tis clear enough the Elephant*
*Is very like a tree!"*

*The Fifth, who chanced to touch the ear,*
*Said: "E'en the blindest man*
*Can tell what this resembles most;*
*Deny the fact who can*
*This marvel of an Elephant*
*Is very like a fan!"*

*The Sixth no sooner had begun*
*About the beast to grope,*
*Than, seizing on the swinging tail*
*That fell within his scope,*
*"I see," quoth he, "the Elephant*
*Is very like a rope!"*

*And so these men of Indostan*
*Disputed loud and long,*
*Each in his own opinion*
*Exceeding stiff and strong,*
*Though each was partly in the right,*
*And all were in the wrong!*

*-John Godfrey Saxe (1816-1887),*
*Blind Men and the Elephant*

# 1 Introduction

*"Water which is too pure has no fish."*

*-Ts'ai Ken T'an*

Increasingly, the physical world, in addition to the people in it, is becoming connected. There are growing numbers of sensors and actuators that are accessible as we navigate through physical space. This includes everything from things we pass, such as bus-stops and stores, to things that sense the environment, such as smog detectors and surveillance cameras, to things that we use, such as taxicabs and printers. We try to design systems that can use the data that these objects generate, and enable us to contribute as well.

Many have predicted that the next generation of the World Wide Web, networking, and communication technologies is going to be about personalization, individual customization, and context. Paraphrasing Marshall McLuhan's "the medium *is* the message" [McLuhan, 1964], if we want to give individual control over the *message*, then we *must* give the individual control over the *medium*. We want to do so without the need to consult experts or an IT team, or to use predefined, hard-coded configurations.

Individuals at the edges of the network need *tools* to pilot, manage, and even comprehend the communication network that augments their social interface to others. In order to achieve ambitious goals of self-organizing and distributedly managed networks, we turn to learn from existing systems that present very similar characteristics to some degree or other – human society. Our approach is to improve the usability of networked devices by modeling their functionality and behavior after observations from the human social realm.

We focus our discussion on the subset of networking that deals with proximate communication, which we dub *"Face-to-Face Networking" (FtFN).* Network interaction in this space is often analogous to human face-to-face interaction, and increasingly, our devices are being used in local situations. We describe existing work in the field, from both social and technical perspectives (***Chapter 2***). The unique characteristics and design challenges that are common to applications for this interaction space, which combines simultaneous communication through traditional human means and digital means, are also highlighted (***Chapter 3***). We then present the main contribution of the thesis, the framework of *Virtual Private Milieus*, which is our proposal for ameliorating the problems described above. A "*Virtual Private Milieu*," or a "VPM," allows networked devices to act on our behalf and project a "digital aura" to other people and devices around us in a manner analogous to the way humans naturally interact with one another. The dynamic aggregation of the different auras and facets that the devices expose to one another creates social spheres of interaction between sets of active devices, and consequently between people. We describe a VPM framework, key features of which include the incorporation of trust and context parameters into the discovery and communication process, incorporation of multiple context-unique identities, and also the support for multiple degrees of security and privacy (***Chapter 4***). Our discussion continues to present the "*Social Dashboard*," a novel human-centric user interface that functions as a readily usable control for one's aura (***Chapter 5***). At this point we expand the discussion to show how our socially inspired concepts can be generalized and applicable to all facets of networking, embodied in the idea of *Social Area Networks* (***Chapter***

***6***). Following this discussion, we move to describe our implemented work. We give an overview of *"Comm.unity"*, a code package developed through this project that encases the VPM framework and implements the principles of Social Area Networks. It allows both application developers and researchers easy implementation of various socially informed applications (***Chapter 7***). We also present two applications developed over this platform, a standalone *"Social Dashboard"* application and *"SnapN'Share"*, which enables users to generate and seamlessly share content with different groups and communities that they belong to, as they come within close proximity of their peers (***Chapter 8***). Finally, **chapter 9** provides a closing discussion of the ideas presented, highlights the work that lies ahead, and peers into the more distant future, a future where we might be able implement all of our acquired knowledge of human society to create a new breed of autonomous network devices that act as a society, are inspired by human society, and are as flexible and robust as human society. For the interested reader, ***Appendix*** section provides a relatively philosophical discussion regarding networks, media, and people.

In our discussion we shall refer to the term ***digital aura***, which we associate with the digital facets of identity and content that are linked to a person at any given moment. The digital aura would includes, for example, a person's current GPS location as he or she moves about, their instant messenger client's online-status, their digital chat messages, and their voice calls. The digital aura would also include their homepage or current Facebook [Facebook] profile, and even content placed in shared directories, open to the Web or to specific peers. In other words, it is everything that describes the current *state* of a person in the digital realm. We can think of the digital *aura* as something that surrounds and *emanates* from a person as he or she roams the physical or digital realms[1].

**Figure 1.1 – Reclaiming Our Digital Aura [Source: "Peter Pan and his shadow,"]**

[1] We use *"digital aura"* rather than *"digital footprint"*, a term that has been associated with the personal information that people leave in the online world - from name, address and phone numbers to text, photos and video that people post in public and semi-public places [Gantz et al., 2008; Madden et al., 2007]. The distinction that we try to make is that the "footprint" would include all traces, logs, or any other information that is accumulated over time and may even outlast the person. The digital aura is something that can be actively managed, rather than a log that is left behind. The two are connected in the sense that the way a person conducts their digital aura, for example what information they choose to expose to third parties, would affect the logs and information that others can accumulate and store about a person outside the realm of that person's control. For example, information "snapshots" generated by Google's cache or Alexa (www.alexa.com).

# 2 Face-To-Face Networking (FtFN)

*"What you see is what you get!"*

*-Geraldine, the Flip Wilson Show*

Many of the ideas presented in this thesis could be implemented in a variety of networking and communications domains. The choice was made to focus on the subset of networking that deals with of personal, close vicinity communications, which we dub *Face-to-Face Networking*, or FtFN. We motivate the reasons for selecting these types of networks in section 2.1. The key motivations are that in this space, the interaction of networked devices is often analogous to human face-to-face interaction. In addition, the FtF domain still has open issues that require solutions, which our approach could help mitigate.

In the framework of our discussion, by "Face-to-Face Networking" we refer to short range, close proximity peer-to-peer networks. Users of these networks are aware that any device that they discover or interact with through the network is physically proximate to them. Phrasing it another way, these networks allow users to interact digitally with the people and devices that are around them. A more detailed definition is given in section 2.2. It is important to note that the notion of FtFN is not tied to any specific technology, but to an architecture that could interface with existing technologies and unify them under one roof of common communication primitives and interaction goals. There are multiple existing technologies that would fit under the umbrella of FtFN, like Bluetooth [Bluetooth SIG] and other technologies, which we detail in section 2.5

Before diving into the technological review and requirements, however, we aim to better understand the human and social side of face-to-face interaction. In 2.3 we present a sociological discussion that examines various aspects of FtF communications in the modern age, through related studies. We demonstrate the enduring importance of FtF interactions. We also examine the type of interactions and who people interact with. We review some of the studies that show how FtF interaction is important for both cultivating our connections to our closest circles of friends and relatives, but also to interact with people that we do not know very well, or not at all. We then mention some of the works that look into the effect of digital communication technologies on our FtF interactions. These effects gain importance as network technologies becomes more pervasive and mobile, and move out of the home and offices settings into streets and public places. Some of the concerns that go along with this increased connectivity relate to the risk of increased "privatism" and "cocooning" effects, where people are more engaged with their remote connections than with the people and objects in their physical surrounding. At the end of section 2.3 we build on these social findings in order to formulate a technological approach and high-level requirements for answering the needs while mitigating the problems in the FtF interaction space.

One of the main goals of this chapter is to discuss the common characteristics of applications and technologies in the FtFN space, and the clear distinction between these technologies and other types. We provide our review in sections 2.4 (applications) and 2.5 (technologies). Finally, in section 2.6 we present the discussion of the motivation for defining FtFN as a unifying umbrella, and for creating two complementary network stacks – one for infrastructure networking, intended to provide a connection to anyone, anywhere, and

anytime, and the other for close vicinity applications that deal with the "here" and the "now".

## 2.1 Motivation: Face-to-Face Networking

**Issues with FtF Digital Communication**

Even though mobile wireless devices have become ubiquitous in the networked society[2], we still use them mostly for accessing the big "network cloud" through cellular base-stations and wireless access points, rather than communicate directly with each other and with objects around us. Infrastructure networking has immense advantages for many purposes and contexts. Unfortunately, it is not always available or cost-effective to deploy. This is true for extreme situations like terroristic attacks, natural disasters, or the battlefield, but also in mundane situations of modern society – in the subway or the basement. A positive point, however, is that in many FtF scenarios the infrastructure is not necessarily needed.

To give an example of this point: With today's common technologies, if I want to give a document file to a friend standing next to me, for example, I would most likely send it through email. This email message would be routed to the at least one server, that might reside on the other side of the world, and come back down to my friend's device. We would be using a service provided by an *external* operator, utilizing *remote* network resources along the route and probably *paying* money for these *services*. There is also a substantial *setup* that had to take place before we could do it, for example: Deployment of the operator's infrastructure to cover our physical location, configuring servers, subscribing to the service, authenticating through an access point, and having prior knowledge of global network addressing (e.g. I need to know my friend's email address). This is done despite the fact that our wireless devices could technically talk to each other directly through interfaces they already have. For example: Wi-Fi in ad-hoc mode [IEEE 802.11], Bluetooth [Bluetooth SIG, 2007] or an infra-red interface [IrDA]. They could do it anywhere, disregarding internet or cellular coverage (or lack thereof), and for free. We could summarize this with the following illustrative analogy: ***Sending a file to my friend through email is like to using FedEx to hand my friend a pencil.*** Sometimes it makes sense, other times it does not. We touch more on the issues with today's FtF networking in section 2.6.

Instead of viewing our devices as remote terminals of the big "network cloud", the goal is to try and make them seem like *natural extensions of our body and senses* - To try and make digital face-to-face interactions with our peers as easy as talking to them or handing them a pencil. Marshall McLuhan sees all media as extension of "Man"[3], but it could also amputate Man at the same time [McLuhan, 1964]. There are many ways to implement this augmentation, and simply extending the existing would never be fully possible, as McLuhan himself says: "Any technology could do anything but add itself on to what we already are."[McLuhan, 1964, p. 11].Our goal, however, is to make the transition of our senses, interfaces, and consciousness from the biological to the digital as smooth as possible, rather than the abrupt "step-function" that it currently is. In this chapter we will propose that at least one way to do so is through our socially aware network tools.

[2] As discussed extensively by [Castells, 1996], [Caron & Caronia, 2007], and many others.
[3] McLuhan uses the capitalized "Man".

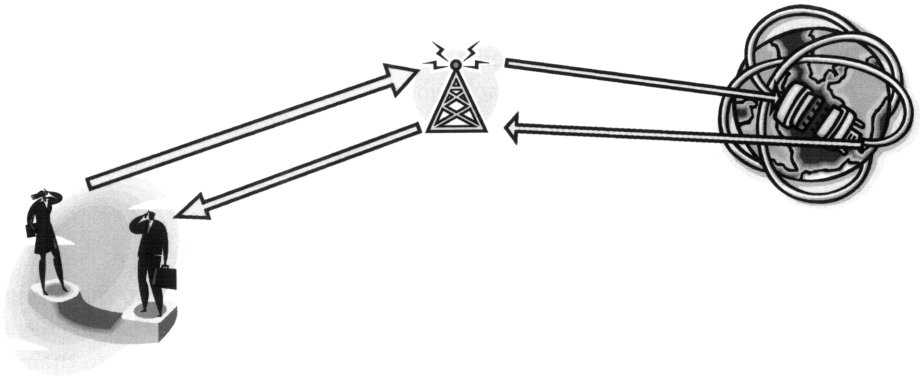


Figure 2.1 – Conceptual illustration of infrastructure based FtF communication.

**Similarity of Human and Digital Interaction in F2F Space**

Solving a real problem is not the only reason for picking FtF networking as our testing domain, or even the most important reason. In order to explore the socially inspired ideas, it was important to pick an implementation realm that on one hand allows us to explore as many of the concept's features as we can, while at the same time minimizing any extra efforts that do not directly support the concept. FtF networking is ideal as the first test scenario for several reasons:

- Analogy: FtF networking is very much analogous to face to face social scenarios: each person or entity in the proximity is represented by a device with a wireless radio. In the same way that people can look around them and use their senses to learn who is in their vicinity and assess the social situation – by seeing with their eyes or listening with their ears – the devices can use the radio medium to scan other devices in the vicinity and assess the situation. In the same way that a person can passively overhear conversations of others and infer social relationships between them, many times we can achieve similar results with a passive radio, listening to exchanges of information in its vicinity.
- Less complexity than other networks: By having the wireless radios in close vicinity (one-hop) of each other, we can avoid the need for complex network mechanisms which make distributed networking so hard to implement – global address allocation, multi-hop routing, and keeping an up to date picture of an ever dynamic network topology.
- Despite simpler technical aspects, still spans complex scenarios: The one-hop network scenario does give us a wide span of networking activities - We can have unicast communication between two parties, multicast to multiple devices in range, or broadcast to all devices in range. We have two or more end devices, communicating with each other through the entire networking stack, from the physical layer up to the end-user applications.

These characteristics make FtF networking ideal as the first testing domain of our socially inspired ideas and the *Social Area Networks (SocAN)* architecture that we detail in chapter 6. As we gain more knowledge about the social metaphor for networking, we can extend the SocAN ideas to other networking scenarios, like multi hop networks that span different locations, infrastructures, and networking technologies. To summarize, these are the reasons that motivate our implementation in the FtF realm:

1. FtF networking is very much analogous to human face to face social interactions.
2. The domain is relatively simple from the networking point of view, while still offering a wide range of network behaviors.
3. FtF networking still has open problems and a successful SocAN implementation could help solve real world problems rather than just a toy problem.

## 2.2 Definition: Face-to-Face Networking

In the framework of our discussion, by "Face-to-Face Networking" we refer to short range, close proximity peer-to-peer networks. Users of these networks are aware that any device that they discover or interact with through the network is physically proximate to them. Looking at it from the other direction, these networks allow users to interact digitally with the people and devices that are around them.

As we mentioned in the opening part of the chapter, it is important to note that the idea of FtF networking is not tied to any specific technology, but to an architecture that could interface with existing technologies and unify them. This is why there is no absolute definition to what constitute "proximity" or "vicinity". Specific distances depend on the specific networking technology in use and other factors like the topography of the physical environment and nearby radio interference. For most cases, we look at scenarios where the users can see their peers and/or their communication devices with their own eyes and senses– for example a person desiring to transfer a file to the devices of all peers present at a business meeting, or a person attempting to receive some digital content from a digital kiosk as she walk past it. Nevertheless, there are other interesting scenarios that we would consider in our definition of FtF networking, like walking down a corridor and being notified that an acquaintance is nearby in one of the adjacent rooms.

Some of the FtFN functionality could be emulated by using a centralized architecture – For example two devices who report their geo-location to a centralized server, which is how the Loopt service operates [Loopt]. We refer to such technologies in section 3.1. This approach, however, is not of much interest to us from the technical standpoint. The social discussion and requirements that are mentioned throughout the chapter are very much applicable to centralized implementations of close-proximity networking. For the majority of our technical discussion, we look at single-hop networks. By single-hop we mean that there are no intermediary nodes between the end points - there is a single transmission from source to destination.

## 2.3 Theoretical Background: The Importance of Being FtF

Before implementing a solution for the FtF space, we want to understand the characteristics of FtF interactions in the modern age, as well as the effect of new media and digital communication technologies on this primal form of human interaction. In order to design the right networking tools, we want to know what kind of interactions people partake in, with whom (Strangers? Close relationships?), and some of the problems introduced with the existing technologies in this space. Much research has been done in this area, and only a few of these works can be sampled in our current scope. We look at studies that start that start from the mega-city scale [Castells, 1996; P. Hall & Pain, 2006] and go down in scale to

neighborhoods[Hampton, 2004], apartment buildings [Foth & Hearn, 2007], small coffee shops [Hampton & Gupta, In Press], and a person's interactions with close family and friends [Caron & Caronia, 2007; Wellman, 2001].

### *The (Un)Death of Distance*

As Internet and remote communication technologies started to gain traction, much was said about the impending "Death of Distance" – predictions that these technologies would "cancel distance", and allow people to use remote communications and remote presence technologies for achieving the tasks they used to do by meeting face to face. However, this is not an absolute truth. Many of the studies noted above conclude that even with the great pervasiveness of the Internet and electronic communication technologies, FtF is still a very important aspect of human interaction in all of those levels and in all social environments that were studied – global corporations, residences, and leisurely places. Wellman notes that on one hand, with today's computer mediated technologies, "Physical closeness does not mean social closeness". However, on the other hand he notes that social relationships actually do benefit from physical proximity, since the interaction with the majority of members in an individual's network tends to be face-to-face [Wellman, 2001].

### *Privatism and the Decline of Social Capital*

In *Bowling Alone*, Robert Putnam notes the decline of "social capital", and finds that people spend increasingly less time with friends, relatives and neighbors, leading to declines in activism and participation in community organizations. Although Putnam notes various reasons that might be causing this shift, he focuses on forms of communications media, mainly television, as contributing to these effects and encouraging people to stay at home rather than do outside activities [Putnam, 2000]. Wellman continues discussing this trend and notes that people are increasingly likely to socialize in small groups in private homes rather than with large groups in public spaces. He analyzes this trend as a change from a social network with many *weak ties* to a network with fewer *strong ties* [Barry Wellman, 1992; Wellman, 1999]. The negative implications of this shift can be inferred from Granovetter's earlier work about the importance of weak ties for people's day to day functionality [Granovetter, 1973]. Wellman's weak ties and strong ties are akin to Putnam's *bridging social capital* and *bonding social capital* [Putnam, 2000]. Putnam expresses the concern that increase of bonding social capital comes at the expense of bridging social capital which coincides with the increased privatization of communication networks. Boase et al. [Boase et al., 2006] note that FtF and telephone contact remain the dominant modes of connectivity when people communicate with their closest ties.

### *An Inconclusive Debate*

After learning this, we would like to have a better low level view – what types of interactions people do FtF, and how has new media affected people's social life and relationships. In recent years there has been an inconclusive debate on the effect of the Internet and other new media technologies on traditional forms of communication, like FtF interaction, and on the traditional structure of communities. Some works, especially from the initial days of the World Wide Web find more negative effects [Kraut et al., 1998; Nie & Erbring, 2002]. Others take more optimistic views [Calhoun, 1998; Hampton, 2004; Mitchell, 1995; Quan Haase et al., 2002; Rice, 2002]. Very similar concerns and discussions came up in the context of the transition from an agrarian preindustrial society to an urban industrial society [Milgram,

1970; Wirth, 1938]. From results of multiple studies, it seems that these technologies hold as much promise for reconnecting people to communities of place, as it does for liberating people from them[Hampton, 2004].

### *Shift to Mobile Communications: From Connecting Places to Connecting People*

*Nothing follows from following, except change.*

*-Marshall McLuhan, Understanding Media (p12)*

What does emerge from the studies mentioned above is that we are living at a time of change. As network technology becomes more pervasive and mobile, our social life does not remain static. Already we can point out a shift from communication between places to communication between people. If in the past we used to call a home phone number and ask *who* is speaking, we now call a person's mobile number and ask *where* they are [Wellman, 2001]. As Wi-Fi and cellular data connectivity gain popularity, it has become possible to integrate intensive Internet use with the use of public spaces, whereas before the use of internet was confined to wired communications from the home and office [Hampton & Gupta, In Press]. This changes the characteristics of physical spaces. Places not designed for communication become new communication spaces – like sidewalks, cars, and public transportation. Part of the risks to FtF interaction is an increased effect of public privatism [Hampton & Gupta, In Press] and the *cocooning* of people into their remote connections, ignoring their face to face peers [Caron & Caronia, 2007, p. 18; Goldberger, 2003; Harris, 2003]. Blanchard looks at this dissonance between the FtF social situation that a person is part of as he engages in computer mediated communication (CMC), and the social situation that is formed through the CMC interaction [Blanchard, 2004].

### *From theoretical background to technical requirements*

A key observation that we make is that all of the technologies studied and currently available for mobile users are infrastructure based, and designed to connect the users, through their digital aura, to remote interactions. We ask ourselves whether the current tools are lacking. Should the tools intended to connect users instantly to the other side of the world anytime and anywhere be expected to improve connections for the here and now? Perhaps there are other technologies that are more adept for the face-to-face realm. We ask ourselves whether the use of peer-to-peer and device-to-device technologies could help deal with the issues that so many of the papers observe or warn against.

Perhaps the reason for some of the reported negative effects on social interaction is partly due to the way our current communication tools work, and the fact that we still do not have the right tools that encourage engagement with our close proximity. For example, we use technologies that take our consciousness and connect it very efficiently to a remote place; We know how to connect instantly to the other side of the world, anytime, from anywhere. Perhaps we should think of technologies that search through our locality before launching us into the remote space, for example.

In this context we can note that the tenants in Foth's study [Foth & Hearn, 2007] had only two options – either broadcast messages to everyone (the open discussion board), or unicast to specific peers, via the instant messenger (IM). This dichotomy between two limiting extremes is very common in many applications and communication technologies. Perhaps if

there was a better implementation that allowed communication between subsets or groups of residents, these participants would feel more comfortable to share information with more people, and make better use of the CMC technologies. We discuss this extensively in chapter 4, as we introduce the details of the VPM framework. VPMs allow for both collective and networked interaction [see Foth & Hearn, 2007]. Users can create their own personal groups and connections, and also register to contexts and groups that are externally generated, for example by the building management, research department, or municipality.

We can see two main needs arising from the collection of works cited above: First, to communicate better with friends and family (strong ties, "bonding social capital"); Second, to meet new strangers with common interests or to accomplish tasks (weak ties, "bridging social capital").

So far we discussed solutions for specific problems. Could we generalize from that to higher level goals and designs? Could we reclaim and repurpose these interaction spaces as places of communication with co-located peers? Could we find a way to break the "privatization" barrier, and allow new social interactions that would not have occurred otherwise? For example, sharing music within a subway car, a new mode of communication on its own, but one that may lead to more general person-to-person interaction. This interaction might evolve from using one's peers as "servers" and "storage", to "hey, we like the same music, maybe we have more in common". Or in short, ***could we find ways that would let our digital auras discover one another and then introduce their owners?*** As Foth&Hearn note, "the networking and linking practices enabled by new media and CMC Technologies are not going to be solely based on co-location and be *about* a place, but on the socio-cultural meanings that people derive from interactions situated *within* place." [Foth & Hearn, 2007]. This is exactly the aspect that we want to explicitly float to the surface with SocAN based FtF networking, under the assumption that making the combination of co-location and social relationships will facilitate inter personal interactions and encourage new ones.

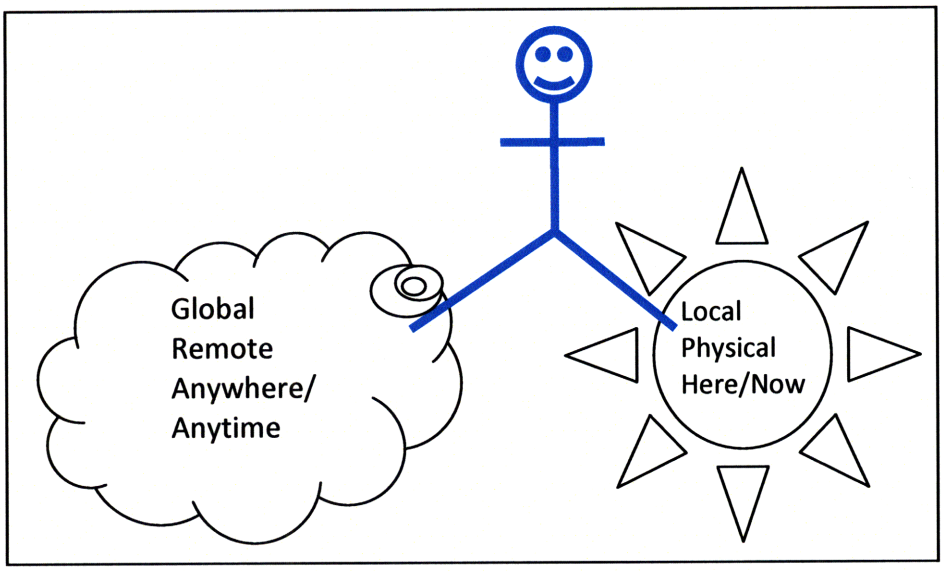



**Figure 2.2 - One leg in the local, the other in the global.**

Finally, a realistic approach should not focus on solely the local or the global – both should be in the mind of the designer. As illustrated in Figure 2.2, today's users live in both the local and the global realm. ***Rather than working separately on technologies that would further separate the two spaces and stretch our poor user's legs until an unfortunate sprain[4], we should build technologies to merge the two realms into one hybrid space, and give our user a stable foothold.***

## 2.4 Related Work: Close Proximity Networking Applications

As mentioned earlier, even though widespread wireless standards like Bluetooth, WLAN, or IR allow devices to talk directly with one another, they are not commonly used this way. Bluetooth and IR are primarily used for communication between a single user's devices (PAN), and WLAN is mostly used to connect to access points in "infrastructure mode" rather than to peer devices in ad-hoc mode. In this section we review some those projects that do fit in the close proximity networking space. As in the related social studies, there is a plethora of work in this area, and we only sample a few here, in order to establish our discussion

### 2.4.1 Close Vicinity Communication and Notification

In recent years there have been several implementations systems that use either device to device communication or allow communication, notification, or content exchange between peers that are in close vicinity. Let us sample some of them: Many applications in this area belong to the category of "location based dating", trying to locate potential mates from other users of the system who are nearby the user, for example Loopt [Loopt], Meet Moi [Meet Moi], or Get Sweet [GetSweet]. The majority of these applications uses central servers and requires prior installation of software and registration. Get Sweet does require a server, but performs its sensing of local peer through a mobile phone's Bluetooth interface. "Cityware" [Jones & Kostakos] is an application that interfaces with Facebook [Facebook] and allows users to get notified when other Facebook users are nearby. The system uses Bluetooth sensing and requires installation of an application. However, the system requires the deployment of local servers that actually perform the device sensing, and one has to be in range of such a server in order to be detected by the system. There are very few such servers deployed. None of these systems are ad-hoc or self-configuring.

The Serendipity project does not require a server to be present in the proximity in order for two devices to detect each other. However, it does necessitate an internet connection. Serendipity uses Bluetooth proximity scans with the goal of actually making new introductions among the system's participants. When two strangers come within range of each other, the application queries a central server that holds profiles of participants. When a match of interests is found, both users receive an introduction message [Eagle & Pentland, 2005]. The "Familiar Stranger" project, [Paulos & Goodman, 2004a], employed radios to sense peer devices in the proximity and generate visual cues. The ideas is inspired by Stanly

[4] Or even worse, a complete tear...

Milgram's concept of the same name [Milgram, 1977]. MyNet [Kalofonos et al., 2008] is a proposed platform for distributed communication and content sharing, that regular non-expert users should be able to utilize and manage. A prototype demonstrating proof-of-concept for some of these features has been successfully implemented and tested. It deals mostly with previously known and trusted devices rather than strangers and although it is secure, it does not deal as with confidentiality issues as our proposed approach. MyNet is currently implemented over the UMI architecture, described in detail in section 4.5, where we also analyze its tangent points to the work of this thesis.

### 2.4.2 Distributed Ad-hoc Wireless Applications

Microsoft's Zune music player offers a feature of sharing music wirelessly with nearby peers. However, this capability is extremely limited, working only with MS Windows machines, and each received song only lasts three plays or three days before it is disabled [Zune]. Another example is the Nike+iPod Sports Kit [Murph, 2006]. The kit consist of a sensor put into a user's running shoe and wirelessly transmits a singal to a receiver attached to the user's iPod Nano. The sensor measurements are used for giving the audio feedback about the user's workout. A key downside to these systems and similar commercial applications is that their implementation usually implements the needs of that specific application, and it is usually not open. It seems that there is room for a software interface that would simplify the creation of device-to-device applications and enable easy creation of similar and new applications. In this chapter we try to make the case that FtF networks are in a different category than the traditional, infrastructure based technologies, and thus warrant a customized toolbox of their own. Hopefully such a toolbox would make the creation and use of FtFN applications as simple as applications for the World Wide Web.

### 2.4.3 FtF Technologies Used for Social Research

As mentioned throughout this document, mobile devices that are carried by the users can be used for learning about their behavior and social interactions. In this section we review a few of the projects in this area. We referred to the general context of using network information for social research as well as the reality mining project effort in section 6.6, so we will refer to additional projects here in the context of FtF.

Meme Tag uses infrared sensors to register other users of the system that come face-to-face with the wearer. It then makes use of prerecorded answers to survey questions that both users answered to flash LED's indicating whether the two have common or different answers and interests[Borovoy et al., 1998]. Meme Tag has evolved to a commercial company offering these and much more advanced services for conferences ["NTag,"]. The Hummingbird project uses custom mobile RF devices for alerting users of the system that they are co-located. The project was aimed at supporting collaboration and augmenting forms of traditional office communication mediums. Users of the system in several conferences and rock festivals noted that it fostered a sense of connection in an unknown situation [Holmquist et al., 1999]. Jabberwocky is a system that uses built in Bluetooth interfaces on mobile phones to scan for peers and visualize these sensed encounters,

helping users to develop a sense of urban community [Paulos & Goodman, 2004b]. The system fosters ideas like Milgram's 'familiar stranger' [Milgram, 1977].

## 2.5 Technological Background: FtF enabling technologies

We shall now look at some of the existing link layer technologies that could be used for implementing FtFNs. We also touch on other techniques, some of them still experimental, that could support interesting FtF networking interactions.

### 2.5.1 Existing Standards

There is a large number of technologies and standards that operate in the close proximity space, some of them overlapping in functionality and offerings. The reasons for this are varied, from technologies that evolved along parallel paths, to market forces and business considerations that led to the development of competing standards. These technologies include:

**Bluetooth (BT)** – Bluetooth wireless technology is a short-range communications system intended to replace the cable(s) connecting portable and/or fixed electronic devices. [Bluetooth SIG, 2007]. BT devices arrange themselves as *piconets*. Each piconet has a master that controls the synchronization of the group and it can communicate with up to seven other devices. Bluetooth specification allows connecting two or more piconets together to form a *scatternet* with some devices acting as a bridge by simultaneously playing the master role and the slave role in one piconet [Bluetooth SIG, 2007; Tanenbaum, 2003].

**Wi-Fi (IEEE802.11)** – the Wi-Fi standard (IEEE standard 802.11) comes in different versions (a, b, g, and others). What interests us most for our discussion is Wi-Fi's ad-hoc mode. An ad-hoc network is a network where stations communicate only peer to peer (P2P). There is no base station (access point) and no one gives permission to talk. Wireless devices within range of each other can discover and communicate directly without involving central access points [IEEE 802.11].

Wi-Fi and Bluetooth are the most commonly available and used FtF technologies. Other technologies, in different levels of adoption, include ZigBee (IEEE 802.15.4) [Zigbee Alliance] , IrDA for infrared short range communication [IrDA], WiMax (which has the longest range out of the technologies on our list) [WiMAX Forum], among others.

### 2.5.1 Experimental and In-Development Technologies

Aside from the technologies already on the market, there are many technologies in development that could add to FtFN functionality and give the users the ability to better control their transfer of information. The following list also includes several projects that are being implemented by the author, but their detailed review is out of scope for this thesis.

***Touch/Near-Touch Data Transfer***

Transfer of information by touching or almost touching two devices together give the user a lot of control over the time and destination of their information transfer. This is also good for security purposes, as it is much harder for undesired parties to "overhear" the information being communicated. This is one of the reasons that technologies of this type have been considered as ways to establish a trusted channel for sensitive information, like encryption keys or establishing initial trust between two devices. A key aspect of this sense of security is that the users see with their eyes with whom the exchange the sensitive information, and use this to augment the digital identification. This approach is used the MyNet project [Kalofonos et al., 2008] and is intended to be an important tool of FtFN. Near Field Communications (NFC) [NFC Forum] is already on the market and is gaining success in Asian countries like South Korea and Japan. However it is still not prevalent in Europe and the Americas. This is probably due to the fact that most of the current commercial usage scenarios of NFC are of the "infrastructure mindset" rather than the peer-to-peer mindset – touching the device to designated artifacts allows the user to pay for a service (e.g. subway), purchase the object, or connect to related information, products, and services through the Internet. This requires a large infrastructure to be set up before deployment. Aside from NFC there are several other technologies and projects in this area. One of them is possibly the newest addition to the communication standards collection, the TransferJet consortium was announced on July 17th [TransferJet Consortium, 2008]. TransferJet is another close proximity wireless technology, that would enable high speed transfer of multimedia for extremely short distances (~3cm). Another type of technology, dubbed at the time "Personal Area Networks[5]", allows using the conductivity of the human body as a data transfer medium under certain conditions [T. G. Zimmerman, 1996].

### *Additional Close Proximity Standards*

Several standardization and industry efforts are underway that would eventually lead to commercial technologies in the FtFN space. WiBree is a new standard that is intended to eventually replace Bluetooth, and has been adopted by the Bluetooth Special Interest Group, and will become part of the BT umbrella. Its main premise – low power consumption that will greatly increase battery life [Foley, 2007]. Internet Zero, or Internet0, is an initiative by the Media Lab's Center for Bits and Atoms in collaboration with a consortium of companies to define a framework to bridge together heterogeneous devices via IP in a manner that is still compatible with designing globally large computer networks. It is intended for connecting consumer devices and other objects to the network, and supports many types of physical interfaces, from radio waves to the electric grid to ultrasonic transmissions [Gershenfeld et al., 2004; Krikorian & Gershenfeld, 2004]. Cognitive radios are an emerging field of research and applications that involve the integration of machine perception into wireless radio systems. The "self-aware" radio would be able to modify its parameters (for example its reception or transmission settings) according to active monitoring of several factors in the external and internal radio environment, such as radio frequency spectrum, user behavior and network state [Fette, 2006; Mitola, 2006]. Cognitive radios focus mostly on the low level radio spectrum rather than the full network stack. They could be relevant to our FtFN discussion since a lot of their behavior and sensing is related to the one-hop networking space.

[5] Although nowadays this term is used for referring to a different type of networks. See section 6.1.

### *Net-Eye Coordination: Physically Controlling Wireless Data Transmission or Sensing*

There are several projects in development that aim to give the users better control of their digital data transfer in the FtF ranges and not limited to near-touch distances. Zigelbaum et al. have developed Slurp, which offers a tangible interface that allows users to manipulate information and move it from place to place with the tangible metaphor of an eyedropper [Zigelbaum et al., 2008]. Slurp uses an IR signal for sensing the user's intentions, and the actual transfer is done in the background. Similar hybrid approaches could be used to develop other interesting interfaces for the FtF space. For example, one of the ideas that we are currently working on is a "data flashlight". The flashlight is composed of two parts – a minimal, pin size IR photo-diode that can detect an IR pulse in a wide angle, and a flashlight like device that the sender carries. The flashlight has both IR and visible indication that allows the user to control the transmission angle visually, similarly to focusing the beam of a real flashlight. It is also possible to add a directional antenna to the WiFi or Bluetooth interfaces directly. A regular single directional antenna would be relatively cumbersome and not fit a mobile, hand-held implementation. However, there are several projects that use an antenna array to achieve a similar functionality, and are able to make it very small in size. These implementation and its likes would empower the users of FtFN with a sort of *"net-eye coordination"*.

### *Improving Control over Device Range*

The Bluetooth (BT) standard defines three types of BT devices differing by their offered range - class 1, 2, and 3, enabling ranges of 100M, 10M and 1M respectively [Bluetooth SIG, 2007]. However, BT hardware manufacturers tend to try and maximize range and most devices on the market are of class 1 or 2. It is actually very hard to find any 1M range devices on the market. While increased range is useful for many cases, there are many scenarios where we want to limit the device's sensing or transmission range. Projects like Reality mining [Eagle & Pentland, 2006], Got Sweet [GetSweet], and Jabberwokie [Paulos & Goodman, 2004b], use Bluetooth for proximity sensing, but the granularity that the default BT allows is very limited. A 10M range can place a peer in a room granularity at best, and in many cases the resolution is only as good as a guaranteeing a wing of a building, without even knowing which floor the peer is at. In open spaces, the range is even larger. Until there is commercial hardware and software APIs that enable granular control of Bluetooth power, researchers and hackers have been exploring different ways to control BT sensing and transmission range. For example, in our own experiments with close proximity BT, we actually removed the built-in antenna of a BT USB dongle to get a range of less than 1M. Some projects attempt to BT signal strength values, but these values could be inconsistent, not accurate enough for our needs (accuracy has a 3 meter error) and not always accessible on all BT devices [Castano et al., 2004]. Another way to implement a dynamic control of BT range would be through modifying the transmission circuit by modifying the antenna structure, adding an attenuator before the antenna, or modifying some of the voltages that feed into the Antenna circuit.

### *FtF Sensing Technologies*

There have been many initiatives that aim to provide sensing capabilities for mobile devices. There is also a trend to distribute more and more of the sensing modalities to the devices themselves rather than centrally controlled objects in the environment. A series of projects

have been using custom made wearable 'sensor badge' technologies, equipping a small device with a collection of sensors, that gather information about their user's environment and interactions with peers. These projects include Meme Tag (X Borovoy XX), Hummingbird [Holmquist et al., 1999], UberBadge [Laibowitz et al., 2006], and the Sociometric Badge [Olguín, 2007]. Jonathan Gips reviews several 'low-level' sensing technologies that can be fitted onto mobile devices and allow capturing of social signals [Gips, 2006].

At the time that these projects were developed, they had to be custom made because commercial mobile devices did not have the necessary hardware or were not open enough for developers and researchers to easily access and control the way they are used. Nowadays, more and more of these sensing modalities make their way onto commercial phones, starting with high-end phones and quickly propagating down to more affordable devices. For example, 3-axis accelerometer sensors are already part of many mobile phones on the market, like many Nokia smart-phones, for example the N95 or Apple's iPhone device. In addition to availability of hardware, the software of mobile platforms like cell phones and PDAs is becoming much more accessible to developers and researchers. Moving to commercial devices allows researchers to reduce their development overhead, and also allows the sensing applications to be available to any end user who has the right device models. Projects like Jabberwoki [Paulos & Goodman, 2004b] and MobMedia [Madan, 2008] have already started using commercial devices for sensing projects. Some projects, like VibePhone [Madan & Pentland, 2006] use the phone's regular microphone for recording sound in the user's proximity and use that for inferring context or social parameters. [Kim et al., 2007] uses the phone's camera to record user's eating habits, but the camera could also be used for tracking of location and social encounters (though that might raise other considerations, like privacy of all recorded parties).There are several innovative ways of using mobile devices for sensing that are still in development but show great potential. The network interfaces' radio signals themselves can be used for human sensing, as has been shown by [Woyach et al., 2006] using stationary access points. Aside from their value as computational social science research tools, many of these sensing applications could be used for the direct purpose of improving the FtFN performance of the mobile devices – giving them a wider range of ways to detect and infer social scenarios to control their own behavior and the information they present to their users.

## *2.6 Discussion: The Need to Unify and Simplify*

One of the goals of this chapter's technological review is to show the plethora of technologies and standards that are part of the close proximity networking space. A second objective is to show how similar the core features of these technologies are from the perspective of both users and application developers. The observation is that these interfaces are all centered on the users, emanating outwards from them. Figure 2.3 depicts a sample of the deployed and in-development FtFN enabling technologies, organized very roughly according to the range of their reach.

Most users are goal oriented. They want functionality as basic as "tell me who is nearby", "let me transfer data from here to there", or "send this file to everybody in the room". Different interfaces could be used to achieve these goals under different circumstances, and many times the goal could be achieved by one out of multiple interfaces. Ideally users should not be concerned with the exact technology by which the goal is achieved, but this is not the case today. Even Bluetooth, a standard that hailed simplicity of operation and 'plug-and-

playness' as one of its main features, has become complicated enough that Best Buy has found it profitable to provide a "Bluetooth Pairing" service to help customers pair their Bluetooth enabled devices, as shows . This is true for application developers as well – who nowadays have to tailor their network applications to each specific technology, even though for them too, much of the needed functionality would be similar.

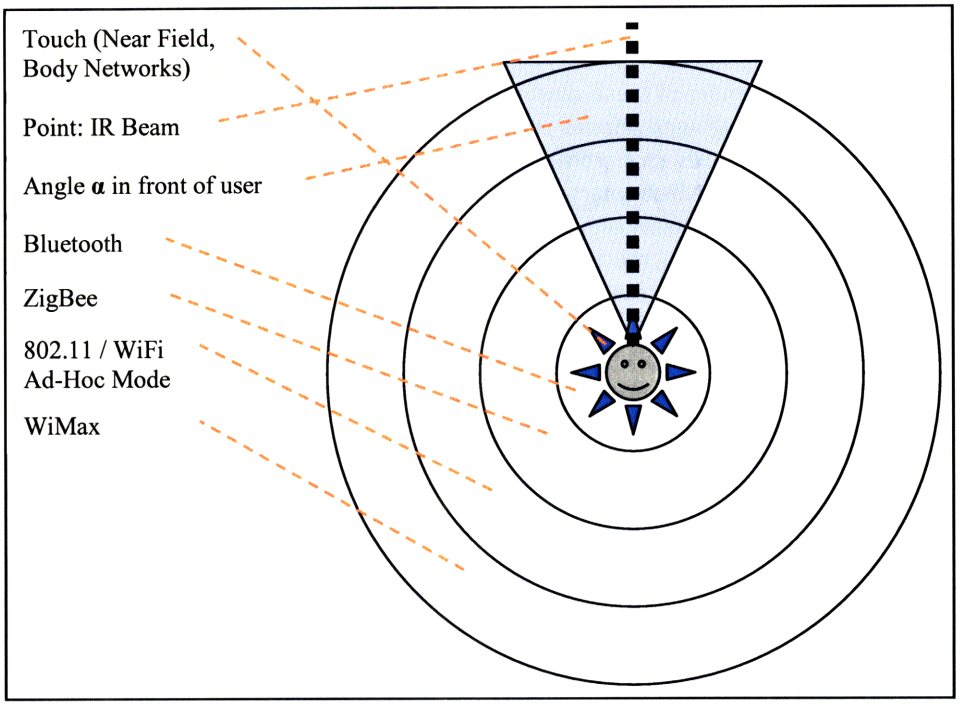



Figure 2.3 - FtF Networking Technologies, centered around the user.

Adding to the complexity is the notion of multi-connectivity – consumer devices have increasingly more types and quantities of physical network interfaces - BlueTooth, WiFi, infra-red, RFID, GSM, satellite connection, and wired Ethernet connection, to name just a few. Many of these radios have FtFN connotation, although by dealing with the issues present with FtFN, infrastructure based interfaces benefit as well.

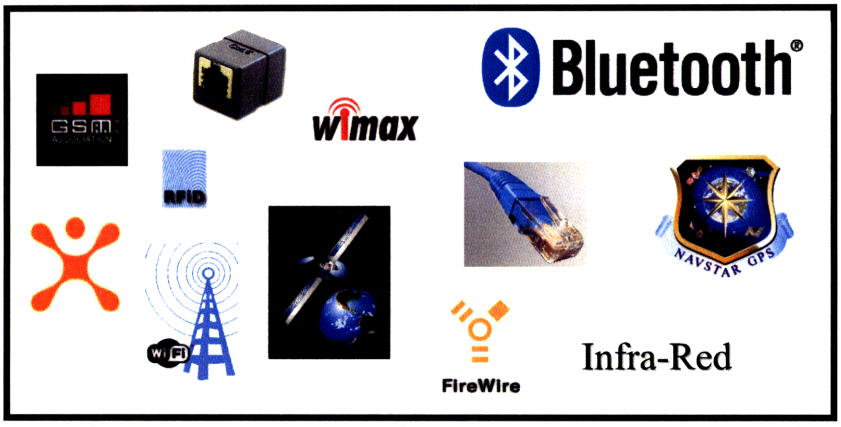



Figure 2.4 - Multi-Connectivity

To complicate things even further, consumers own more networked devices and keep them in close vicinity more often – many people nowadays walk around with a cell phone and a laptop, or a cellphone, a laptop, *and* a PDA. Even though users might want to perform a task that is not FtF in nature, like connecting to the Internet to send an email message, it might be worth their while if they were able to use FtF technology to connect to one of their other devices, and connect to the Internet through it.

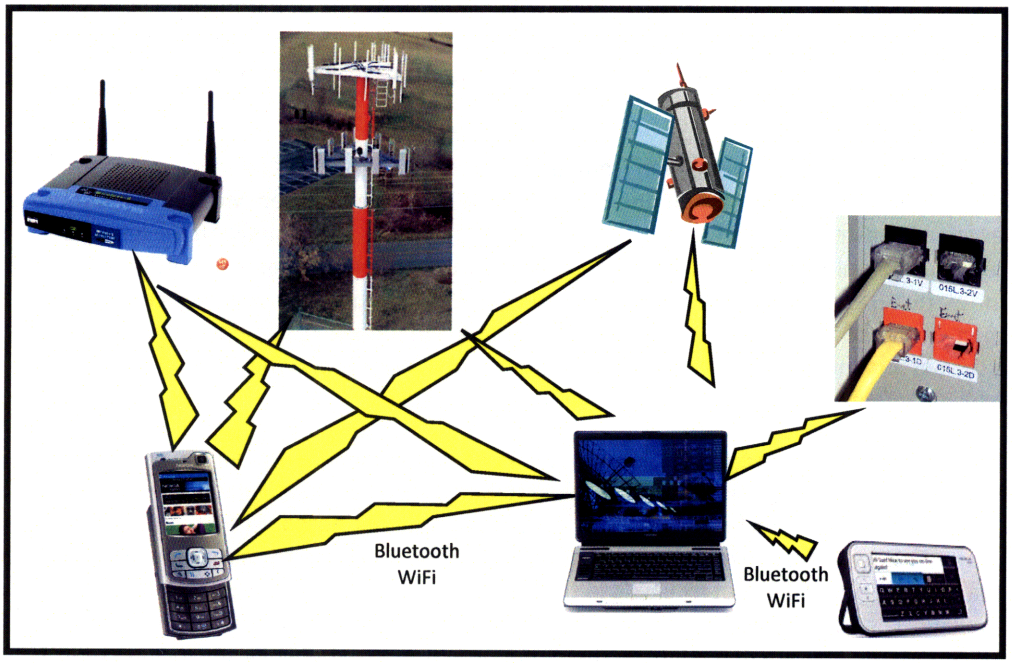



**Figure 2.5 - Which way should my email go?**

Figure 2.5 shows an example of several devices that a user might own and their interfaces to the outside world. An email message sent from the user's PDA device can be sent out in many different combinations. To demonstrate just a few:

- PDA → BlueTooth → Laptop → Wi-Fi → world
- PDA → IR → Laptop → Wi-Fi → world
- PDA → IR → Laptop → Bluetooth → cellphone → world

Which way should we choose? There are many factors to that, such as available bandwidth, speed of connection, cost of connection, device power, and security and privacy issues. Should we encrypt the data? Which access point should we connect to? As we discuss in section 6.3, even expert users do not make an optimal or even a close-to-optimal use of their network resources. In nearly all cases default settings are used, which usually amounts to something like "select the access point with the strongest signal". Other options include using the multiple devices and interfaces in a symbiotic, hybrid way. For example, splitting tasks between different network interfaces to deal with the bottleneck of the wireless medium - for example – use Bluetooth for device discovery but Wi-Fi for data transfer, or use IR to limit range of destination selection and Wi-Fi for the data transfer. Other options are relevant to security – users could define that certain types of data would only be

available on certain interfaces, say near-touch interfaces, while other types of content would be served to the widest range possible. Sensing applications could use the different characteristics of each medium for inference that would improve sensing of peers – for example, if a peer is discovered by Wi-Fi but not by Bluetooth, it might mean that the peer is, with higher probability, located farther away than a peer that is discovered by both interfaces.

To summarize this part, a unifying FtFN abstraction layer would help make this space much more manageable to both users and developers, and this is one of our main goals as we set out to implement a FtFN supporting framework in chapter 7. This platform would also allow for development and testing of more intelligent algorithms to make better use of the variety of devices of interfaces.

Expanding the scope of discussion from just FtFN technologies to networking in general, we can say that up to now, FtF and one-hop networks have been treated with the same tools as existing infrastructure networks – same network layers and a very similar mindset. One notable example is 802.11's ad-hoc mode which was implemented as a spin-off from infrastructure mode. It also exemplifies bad implementations in this space since there it has many issues of security and sustainability (splitting and merging of ad-hoc networks), and issues of and interoperability between vendors and operating systems drivers, many of which implemented the ad-hoc mode with slight changes.

We try to make the case that FtFN is a networking domain on its own, and that Wi-Fi's ad-hoc mode is more closely related to Bluetooth (BT), Near Field Communications (NFC), and other FtF technologies, than it is to infrastructure based Wi-Fi that uses stationary access points. There are many aspects of the multi-hop and infrastructure networking world that are not needed or not available in the FtF world – for example: Routing, DNS, always-available rendezvous servers (or always-available servers in general) and service providers. FtF Standards that are derived from their infrastructure-based siblings and include tools to support these features make the FtF implementation much more sluggish and complex than it ought to be. Second, there are some networking aspects that do exist in both world, but have very different requirements and implementation details. For example – security functionality cannot rely on encryption key-servers, but can benefit from visual authentication by users, making it harder for malicious attackers to masquerade as others. The same goes for functionality like discovery or self configuration of the network. Finally, there are other features that are unique to the FtF space and do not exist in infrastructure based networks, like using devices as sensors and other features discussed in detail in section 2.5.

These observations, combined with our examination of the common requirements of existing applications for the FtF space, lead us to define a middleware layer for FtFN application that is intended to unify implementations of FtFN applications in an interface agnostic manner. This is implemented as part of the Comm.unity platform, described in chapter 7. We hope that this middleware would provide a toolbox that would make the creation and use of FtFN applications as simple as creating new applications for the World Wide Web. Ideally, our devices would be designed from the get-go to live in both spaces - the infrastructure and the face-to-face realm, supporting both types of communication stacks, as depicted in Figure 2.6, which is analogous to the human aspects depicted in Figure 2.2.

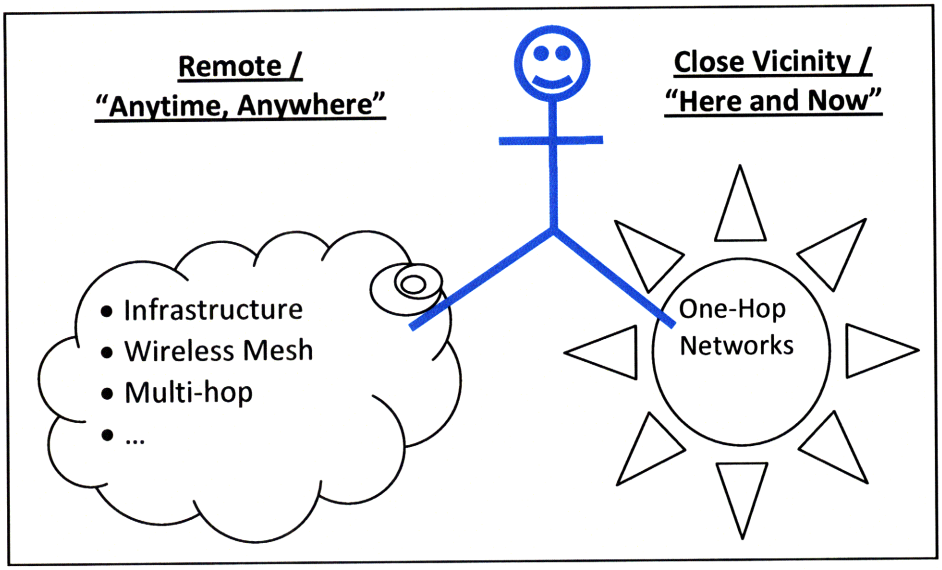



Figure 2.6 – One leg in FtF networking, one leg in the global networks.

## 2.7 Chapter Summary

This chapter was dedicated to understanding the current state of close proximity, or face-to-face, communication in both the sociological and technological sense. From the human side, we tried to highlight the characteristics, needs and potential caveats of FtF interaction in today's highly mobile and networked society. From the networking perspective we argue for the need to unify the handling of close proximity communication technologies under one FtFN abstraction. This argument was based both on the common characteristics of these technologies, together with their distinction from other network architectures and scenarios. We use the social discussion to help us frame the requirements and services that an FtFN abstraction layer should support. As we go forward, towards implementing networking applications and functionality in this face-to-face interaction space, it is very important to be aware of new interaction characteristics that arise from this combination of both traditional interaction in physical space and parallel interaction in the digital realm. We continue the discussion of this new "hybrid" interaction space in the next chapter.

# 3 The Hybrid Space: Interaction on the Seam between Physical and Digital

*"Pay no attention to that man behind the curtain."*

*-The Wizard of Oz movie (1939)*

**Figure 3.1 - Discovering the Man behind the Curtain. Source: ["The Wizard of Oz movie," 1939]**

*"I thought Oz was a great Head," said Dorothy.*

*"And I thought Oz was a lovely Lady," said the Scarecrow.*

*"And I thought Oz was a terrible Beast," said the Tin Woodman.*

*"And I thought Oz was a Ball of Fire," exclaimed the Lion.*

*"No, you are all wrong," said the little man meekly. "I have been making believe."*

*Excerpt from "The Wonderful Wizard of Oz", by L. Frank Baum, 1900.*

Although the Wizard used his multiple personas for deceptive purposes, there is role-playing and presentation of different facets of a person's identity that is part of everyday life and crucial for a society to function. The separation of how we present ourselves at work vs. at home, for example. Throughout the remainder of this chapter as well as the next chapter, which deals with our low level architecture for FtF interaction, we give additional examples of these types of everyday aspects of role playing. For the entirety of this document, it is important to remember that we are interested in *those* types of social interaction.

Let us look at a hypothetical scenario where the Wizard of Oz decided to project his different personas remotely, through the digital realm. There are many ways he could do it - for example by registering multiple email accounts, creating different instant messenger (IM)

usernames, or opening up several separate Facebook profiles. Had he done that, it would have been relatively easy to use existing mechanisms for keeping a strict segregation among the different digital personas and between them and his real-life identity as well.

However, we could also look at the story as a close-proximity interaction scenario, where the Wizard carries a wireless device running a program which attempts to discover the identity of peers around it, and project a different persona to each of them. Theoretically, rather than meet with the four protagonists one after the other, he could have concurrently initiated a different chat session with each of them, each with the respective identity that he wished to project to them. The original story, presents a scenario that is more analogous to such face-to-face networking scenarios than to traditional infrastructure interactions. All parties of the communication physically inhabit the same local vicinity. In the story, the characters used physical cues to discover the real person behind the projected aura. Close proximity digital interaction can lead to similar effects. The network device and the users act as a physical anchor to the projected aura. The aura emanates from them, moves along with them, and is not present when they are not around. The user's physical signals and cues can affect the digital interaction, and vice versa. In extreme cases, this link of physical and digital information could lead to physical danger for the user[6]. In other cases, it might lead to social awkwardness[7]. In this chapter we go into these aspects of interaction, which both users and developers should be aware of.

The previous sociological studies that we reviewed in section 2.3 have mostly dealt with technologies that are already commonplace. In section 2.4 we reviewed novel applications, many of them still not prevalent but they definitely show a trend: After the Internet, "Web 2.0", and the mobile age, we are currently on the verge of another revolution that has already started to seep in. Online presence and social networking applications have already started to merge with our real-world presence and relationships. Although this trend is still nascent, from looking at the growing number of startups and projects in this area we can only speculate that it will greatly strengthen in the near future. Understanding this welding of the virtual and the real world is critical for our design of FtF Networking technologies, but it is also relevant to other infrastructure based implementations, as we shall see in this chapter. For brevity we will refer to this interaction space as the *hybrid interaction space*, or *hybrid space*.

The hybrid space is the unison of Castell's space of places and space of flows [Castells, 1996]. This effect is very powerful: Since our physical presence determines so much of our life's context, this connection facilitates the extraction, at any given moment, of the context-relevant information out of the vast expanse of the digital realm. For example, we could discover that a neighbor has a blog about the neighborhood or our community, or that a stranger on the train shares similar interests to us. In an emergency, authorities could broadcast critical information to people in the relevant areas. It would have been much harder to find these through a global Internet search, and that would also require the user to be very proactive. The hybrid space interaction, however, makes use of the physical proximity as the context for initiating the digital interaction.

[6] For example, when a person's physical location is exposed to a stalker.

[7] For example, when a person's sexual preference profile is inadvertently exposed to one's friends during a night out.

However, this does not come without a price. Touching again on McLuhan's concept of media's extension and amputation of Man - any extension of one sense brings forth an amputation elsewhere [McLuhan, 1964]. At least one of the amputations of the hybrid space is that while engaging in it, a person is neither fully engaged with the physical nor with the virtual. He will become less attentive to his physical surrounding, and at the same time he will also give up some of the freedoms of time, space, context and identity that the pure virtual space allows for. As we mentioned earlier, in the hybrid space, no matter how much he will try to detach it, the digital aura of a person is still bound to his physical presence, as Peter Pan's shadow binds to his feet. It might be possible to blur the connection between the person and his virtual shadow with various techniques, but there is always the risk that someone might trace the digital aura to its physical origin, as in the example of the Wizard of Oz.

We now continue to a review of some of the preliminary applications and technologies for this realm. We then discuss properties of the hybrid interaction, and try to convince that it warrants a different treatment than is usually given to the digital space. Finally we propose some very preliminary principles to keep in mind while designing new applications for the hybrid interaction space.

Figure 3.2 - The Seam between Physical and Digital. [Source: D. Hall].

## 3.1 Supporting Technologies: Absolute and Relative Location

One of the most important enablers of this connection between the physical and virtual interaction spaces is the availability of user location data. There are different location technologies that allow for varying degrees of accuracy, although a high accuracy is not necessarily needed for location aware social applications. There are two types of location information that are relevant for our discussion:

- Absolute location – By using GPS coordinates or by being able to detect that the user is at a specific absolute location (e.g. "The Muddy Charele's Pub").

- GPS: Companies like loopt.com [Loopt] use GPS enabled phones to allow users to share their location with friends, and also recommend events and locations in the physical world.
- Proximity to known infrastructure nodes: As part of the standard wireless communications protocols, there is information about cellular base-stations or wireless access that are in the proximity of the user's device. These can be used to points to triangulate a user's physical location. Googe's new "My Location" service is an example of this type of technology ["Google My Location,"].
- Manual entry: The most basic way, though probably the most demanding from a user's perspective, is for the user to manually enter his or her location to the system, which will then use it to distribute. Services like Facebook ["Facebook,"], Jaiku ["Jaiku,"], Twitter ["Twitter,"], and Dodgeball ["Dodgeball,"] use this type of technology which has the lowest technical entry barrier.

- Relative location – Where the location information indicates relative proximity between users, without the need to know or disclose the absolute geographic location.
  - Device-to-device sensing: Wireless devices can sense IDs of other devices in their range, and infer their proximity according to the signal strengths, packet timers, or use a very simple Boolean detection – if I sense a device, it must be nearby. This uses the logic that the wireless range of the device is limited, so any device sensed is limited within a certain sphere around the sensing device. One example where this methodology has been used in the Reality Mining project [Eagle & Pentland, 2006], where Bluetooth sensing was used to infer proximity between users.
  - Proximity to third party: Some projects have used stationary nodes as mediators between users. A stationary node acts as an aggregator for presence and proximity information, and distributes it to the users. Cityware, a research initiated Facebook application, uses stationary Bluetooth nodes to sense users of the system as they pass near the node, and relay this information to other Facebook users of the applications that are nearby [Jones & Kostakos].

Understanding how these technologies work is important because it could have implications about the user's signals and cues [Donath, Fourtcoming], privacy issues, and the distance (physical or digital) that a user's information can spread (or leak).

Another relevant mention is the Fire Eagle project by Yahoo Research. The service aims to aggregate any type of location information and allows application developers to use location information at any given resolution for their applications [Yahoo Research, 2008]. Although it is not relevant to our distributed, ad-hoc FtFN approach, it could prove useful for applications and scenarios that can afford a centralized architecture and the constraining requirement of an Internet connection.

## 3.2 Applications for the Hybrid Space

A wide range of location based services (LBS) could be built with the help of this location and proximity information. For this discussion we are only interested in social applications, some of which are already out in the market. For example:

- Proximity based dating and introduction services, that match users based on their profile and notify based on current proximity, like those offered by [Loopt], [Meet Moi], or [Eagle & Pentland, 2005].
- Connecting to online content sharing sites, like Yahoo! Research Berkley's Zonetag [Yahoo Research, 2006], which enables location based tagging and sharing of flickr images.
- Allowing family members to stay in touch and let eachother be aware of their location, like Verizon's Chaperone service that allows parents to keep track of their children and receive alarms when they go outside the boundaries of a predefined zone ["Verizon Chaperone,"].
- Applications that allow mobile users to interact with their existing online social networks, like Facebook [Facebook] and MySpace ["MySpace,"].
- Applications for creating new social networks, possibly built on social proximity, like Dodgelball ["Dodgeball,"].

It is my belief that this interaction domain brings new characteristics and challenges. A careful analysis of the signaling aspects in this domain is important as part of the design of an appropriate interaction methodology for these types of applications. From a first glance, it seems that most of these initial implementations do not deal well with the control over presence, privacy and identity information. Mobile extensions of Facebook and Myspace, or of instant messenger applications [Google, 2008; ICQ, 2008; Microsoft, 2007] seem to simply be a limited version of the original applications, adapted to run on the smaller screen of a mobile device.

## 3.3 Characteristics of the Hybrid Interaction Space

In "Deception and design: the impact of communication technology on lying behavior" Hancock et al. develop a framework for analyzing the effect of communication technology on the production of lies. They provide several axes for analysis, related to characteristics of the medium [Hancock et al., 2004].

When we consider the parameters defined by Hancock et al., we can say that the digital or online aspect of a system that is part of the hybrid space has high recordability. Messages between users can be logged much like email or IM histories. As far as synchronicity of the digital interactions, this depends very much on the application itself. It seems reasonable to think that applications in the hybrid space will not fully engulf the user in the digital space, like a real-time video conversation might. It is much more likely that applications developed for the hybrid realm would be asynchronous or semi-synchronous. For this discussion we could characterize them with IM-like synchronicity (see [Hancock et al., 2004]). We are also going to assume that user messages relevant to this space would be short and succinct, rather than lengthy messages or messages that would require long response cycles. Another key aspect is that the online portion of hybrid applications enables the design of applications that allow multiple identities for each person, much like opening different accounts on

today's social networking sites. These could be limited by design (e.g. limiting one user ID per physical device ID), but this might not be desirable, as we'll see later when we discuss the suggested design approach. The reliability of these IDs also depends on the system's design, which can enforce grounding the virtual identity with a real identity, or it might allow any avatar identity to be created within the system.

Because of the direct linking of the virtual application to a person with a mobile device that is present in the physical space, there are additional characteristics that are not part of in the pure digital realm. For example, because the online presence is anchored by a physical presence, there is at least *some* aspect of real identity in this interaction. At the extreme case, for example two people standing alone in the desert with no other device in range, any virtual identity that is detected by one of them *must* belong to the other person standing next to them. Both users as well as designers of applications for this space must at least be aware of this issue.

Another feature added by the physical proximity is the potential of adding real facial and other gestures when there is eye-contact between the communicating users. This might lead to interesting interactions that are not available in regular online chat, for example. From one side, the digital chat may help deal with issues of embarrassment of initiating a face to face talk with a stranger, yet from the other side, the communicating parties may exchange glances to signal interest, or smile in real life to indicate a joke in the digital message.

We can see that this merge of online and offline brings some of the best and some of the worst of both worlds. Some of these characteristics may be considered either positive or negative, depending on the user's intentions and goals. In addition to borrowing from both the physical and digital realms, the hybrid interaction space also brings new features and characteristics that are unique to it, which might raises interesting questions for designers as well as sociologists interested in this area. Following are some examples of interesting questions to consider:

- What effect will it there be to have an online interaction with someone that you *know* must be somewhere in your physical proximity?
- What is the "distributedness" (as defined in [Hancock et al., 2004]) of speaker and listener in this domain? Are they partly distributed? What about situations when the two sides of a conversation share the same space but are not visible to each other? Or if they are sitting back to back and do not see each other's physical gestures?
- How would the users grasp the scope of *public* interactions, for example posting a public query to people in the physical vicinity, or displaying one's public profile? These would be defined in a different way when compared to traditional public interactions in each of these domains. When comparing to traditional public interactions on the internet which are not limited by any physical boundary, these interactions are limited in their extent and range. People on the other site of the world, or even on the other side of the block, may not be exposed to public messages that are set to be spread to only a limited proximity. On the other hand, when comparing to traditional face to face interactions, which are usually limited to the people that are in one immediate proximity and scope of vision, these new interactions could have a much greater span. A person's digital "shout" may be received by people that he or she cannot physically see.

Even when interacting with one's known social network, there might be unexpected differences from what we have become accustomed to. For example, people have become accustomed to update very fine grained status messages in Facebook [Facebook] or Twitter ["Twitter,"]. In the hybrid space, updating one's Facebook status message to "Eating cake" could cause their office to be stormed by hungry acquaintances asking for their share of the pie...

Another example which might lead to awkward situations is setting off a proximity notification between a person and their boss or academic advisor while both occupy adjacent bathroom stalls.

## *3.4 The Hybrid Space from a Signaling Perspective*

For this discussion we are again going to assume that the digital applications that would be available in the hybrid interaction space would, at least initially, provide similar types of media and interaction that current Internet applications offer, like IM clients, online social networks, blogs and forums. In this section we try to analyze them from a social signaling perspective.

### 3.4.1 Signals and Signaled Properties

When we look at the types of signals that are available in this interaction space, the discussion can again be divided to signals in the digital and physical domains.

In the digital domain, there is a great deal of signaling that can be sent out, again depending on the system's design affordances. The notification of one's geographic location and proximity to others is one of the most basic signals that are given off. From location and proximity information one might infer their peer's current context, for example (home/work/bar). Another type of signals is the online presence information that the user may choose to display to peer or different groups of peers. These states could be similar to instant messenger presence information, like "online", "available", "away", "busy", and so on. The displayed identity and user profile are also signals, as well as the affinity groups that one belongs to and chooses to expose. These last ones are mostly from the type of *conventional signals*, where the user can explicitly state the property that they would like to signal [Donath, Fourtcoming, p. 7].

Exposing one's true location could be used to signal trust: "I am exposing you to where I am and what I am doing". Location as well as the conventional profile and group information can be used to signal affiliation and intention. They could signal which groups the user belongs to, which locations they go to. If the location and proximity signaling is done *automatically* and this is known to all participants, this would increase their signal *reliability* [Donath, Fourtcoming, p. 3] over manually entering them, or over the conventional signals. These tools could also be used to signal social stature, by showing that one belongs to limited-access groups (such as the Free-Masons) or hard-to-access locations (like emanating one's digital aura from the hottest club in town).

Signals in the physical space augment the digital ones. They can affirm their reliability, for example when actually seeing a person at the location they claim to be. The merge of physical and digital adds more intrigue to the signaling, like in the example from above

where facial gestures augment a digital chat. Physical actions could affirm or contradict digital actions, and that could be a signal by itself.

Another signal might be the simply the display of a user physically interacting with their mobile device and the system. In the traditional analysis of interacting with mobile Internet devices this might be considered as social cocooning, since the other side is assumed to be in a distant location. However, in the hybrid space this might signal to others that they should also take out their device and see what is happening in the digital domain that's associated with this physical space. It could also signal boredom or detachment from activities occurring in the physical space (e.g. students in a lecture all using the system to chat among themselves instead of listening to the teacher).

### 3.4.2 Cost of Signaling: The Gay Soldier Problem

The users of the system might risk exposing their identity and other private information to undesired parties. In the pure online world, it is easy to own and manage multiple identities, each for interacting with a different group. These identities could be active simultaneously in different windows of one's browser, and there will be no crossover of information between the different communities that the person. However, once we tie the online presence and identity to the physical domain, there might be added risk. An extreme example to make this point could be that of a member of the US military who is also gay. When going out to a pub, for example, this person might want to use the system for being notified when any of his or her Army buddies is around. However, they might also want to use location-based gay dating service. That person would probably want to keep these two aspects of his or her life separate, but may put themselves at risk that their army buddies may find out about their other affiliation. This example shows how the physical part of this interaction space could increases risk to participants or cause damage to their social life.

Many of the current discovery services for ad-hoc networks, especially those that aim to introduce between stranger-devices, usually do not consider such a group-based or context-based segregation, and advertise all available services and identities to all peers in the vicinity. Zeroconf [Williams, 2002] is a widely deployed example of this. Our socially inspired approach, and specifically the VPM framework, aims to deal with such issues directly. We discuss these issues in greater detail in chapter 4.

Additional risks of signaling one's location could be exposing someone to risk of physical harm. Most parents would probably want to make sure that their children's location is not available to anyone but them.

Another type of risk that could result from careless design is exposure of undesired relationship and trust information. For example, imagine a scenario where one participant could infer how another participant views their relationship: "On my client you are marked as offline but I see you here using your laptop for chatting with other people!" Different actions that one can get away with in the online world or traditional physical interactions (where private trust information is only in the participant's minds and not exposed to others) might require more attention in this new interaction space.

Finally, there are also costs of time and effort in configuring all of the trust and privacy settings in such a system, and in manipulating these settings in real time. A bad design would lead to the system being too cumbersome. For example, Loopt's presence interface allows to send one's presence to all of their contacts, none of them, or to specify each contact explicitly (by marking a checkbox). I think that this is going to be too cumbersome to manage dynamically, changing the presence information as users move to different locations and between contexts. I assume because of this cost they would mostly choose to either let everyone know of their location, or none. Another aspect that is relative to cost and time, is the graphical layout of the interface. If the system's settings are hidden behind a series of menus, or if the interface gets overloaded when the user has many contacts and affiliations, the cost might deter people from using the system.


Figure 3.3 - G.I.Joe's Sergeant Slaughter[8]. Source:["SGT2.jpg,"]

### 3.4.3 Deceptive Signals

A benefit for deception and a motivation to do it is the attempt to hide affiliations that one does not want to expose. This might lead users to create fake identities. Such identities could also be used to communicate undetected with peers that do know the user in real life. Deceptive signals might also allow a user to expose the location of their peers without exposing their own location or intentions. From the deception aspect, many other benefits and costs associated with the regular online world that we discussed in class are also relevant here – like issues related to reputation, monetary fraud, creating a fake persona to affect other's impression, and so on.

[8] Disclaimer: Image is for illustration purposes only. The author does not allude or imply that Mr. Slaughter, A Real American Hero, is of Homosexual Inclination

## 3.5 Discussion: Preliminary Design Guidelines

This merge of the online and physical realms has the potential to be very useful – allowing us to meet new people with shared interests, being notified when our acquaintances and loved ones are around, giving us access to desired information and content, and in general - local context is very powerful in many domains and could be used to filter out relevant information out of the endless amount of content in the Internet. However, hasty design and implementation could damage trust and privacy, and even expose users to physical dangers. Alternatively, it could greatly limit the usage scenarios of these technologies, similar to what happened with video-conferencing, which might have made many people feel too exposed to the other side and thus did not use this medium as much as was initially expected. Some initial design principles that will hopefully help us highlight the good and reduce the bad:

- Allow users to create multiple identities and choose how to expose them. It is probably a good idea to make a common fact that each person can have and does have many identity facets in this interaction space. Fist, this would make it so that it would not be instantly considered a deception to have more than one identity. Second, it would make people more scrutinizing about the identities that they interact with.
- Define discreet levels of trust that users could set to each peer or group that they are associated with, allowing granular exposure of profile details. Default should be at the lowest level of trust, and any deviation from it should be explicit by the user.
- Define different types of user group, for example:
    - Public: All can see group content and members.
    - Private: Invite only. Even group name is not exposed to people who are not part of the group. They might know a group is present in their physical space, but not be able to see its name. They will also have no idea how many users who belong to that group are present.
    - Open Sign-up: Group name is visible, but sign-up is required to see its details and associated users.
- Optional: Correlate between trust levels and presence information. By correlating between trust levels and presence information, we might avert some of the awkward situations of exposing relationship trust information unintentionally. Instead of having a peer notice that we are slow to reply to him or need to approve his messages before reading them due to lack of trust, he may just see us marked as "away" or "busy", or even "off-line".
- By default: Symmetry in information exchange. The default state of content exchange should be symmetry. If someone can see a user, the user should be able to see them as well. This issue is mentioned in [Raento & Oulasvirta, 2005] as they discuss privacy management for context awareness applications. Users could opt-out of that symmetry manually, by proactively deciding to expose information to the public or to specific peers.

The next two chapters dive into turning these design principles into an implementable architecture. Chapter 4 deals with the Virtual Private Milieus framework, the engine "under the hood" that supports the FtF networking interaction goals combined with these design principles. Chapter 5 deals with our approach for a socially-aware human-centric user interface.

# 4 Virtual Private Milieus (VPMs)

*"It is not the consciousness of men that determines their existence, but their social existence that determines their consciousness."*

*-Karl Marx, 1859*

This chapter presents the *Virtual Private Milieu framework*, or *VPM* for short. It is one of the key concepts of this thesis, and represents the culmination of the ideas presented in the earlier chapters. The main idea behind the VPM framework can be summed up in one sentence - Just as people present different facets to different peers, we create a mechanism to allow network devices to do the same – according to groups, identities, levels of trust, physical location and social context.

A "*virtual private milieu*," or a "VPM," allows networked devices to act on our behalf and project a "digital aura" to other people and devices around us in a manner analogous to the way humans naturally interact with one another. The dynamic aggregation of the different auras and facets that the devices expose to one another creates social spheres of interaction between sets of active devices, and consequently between people. We refer to these spheres of interaction as "shared settings", inspired by terms used by social and psychological studies regarding human interaction. We borrow additional terms from these humanistic realms both to build on their observations as inspiration, and also to help amalgamate the human and the technical discussions.

VPMs allow users to discover and communicate with peers and groups that they belong to in a secure and private manner, exposing only the desired facet of their identities to each of their peers, all managed by users in a distributed manner. These peers and groupings do not have to be only human, but also standalone devices and services. In more technical terms, the goal of the VPM framework is to provide a robust solution for an unmanaged, group-based discovery and communication architecture among entities with access to a shared communications medium. Key features of this framework include the incorporation of trust and context parameters into the discovery and communication process, support for multiple, context-unique identities, as well as support of multiple degrees of security and privacy configurations.

This chapter opens with a review of social and behavioral psychology studies that explore the way individuals view and manage their social interactions (section 4.1). Next, we present the main concepts and definitions of the VPM framework (sections 4.2 and 4.3 respectively). We move on to discuss the implementation of the VPM framework (section 4.4). Finally, we close with related networking work and technological building blocks that could serve the VPM implementation. These show that many of the tools that are needed to construct the VPM framework are feasible or already exist (section 4.5).

## 4.1 Background and Discussion: Social Interaction from the Individual Perspective

In order to achieve the ambitious goals of self-organizing and distributedly managed networks, we turn to learn from existing systems that present very similar characteristics, to some degree or another – human society. As discussed earlier, our approach is to improve the usability of networked devices by modeling their functionality and behavior after observations from the human social realm. Specifically for developing our idea of Virtual Private Milieus, we want to deal with functionality related to aspects of discovering who is around, assessing the social situation, or context, and deciding on the proper social protocols and information we would expose about our identity and about ourselves in general.

### 4.1.1 The Dramaturgical Approach and Behavior Settings

People follow diverse rules of conduct and have different expectations in different situations. A situation is defined in part by the physical setting - a playground, say, vs. a church or a baseball game; it is also defined by the people there. Sometimes, just their number is important - lots of kids waiting for the slide means everyone must queue; sometimes role is important - classroom behavior is different when the teacher is present vs. out of the room; sometimes individual identity is important - kids behave differently in the same classroom in response to different teachers. We would like our framework to be flexible enough to support similar characteristics. Sometimes just the number of peer devices in the vicinity matters, for example when a Wi-Fi device is trying to gain access to a server. When too many devices try to transmit aggressively at the same time, there is high probability that their transmissions will collide and the data will be lost for all parties. Sometimes, the role is important – a server, an access point, a firewall - and it does not matter for the situation which specific device fulfils that role. Sometimes just the differentiation between an unfamiliar "stranger" device and a familiar device would call for a different behavior, for example, a stricter authentication process for strangers. And of course, many times the individual device does matter. *My* laptop may have an intimate relationship of trust and mutual sharing of services with *my* mobile phone, but no other mobile phone that it may come across.

Different terms are used in the sociological and psychological literature to represent these notions of behavior, social situations, and identity. Erving Goffman is best known for his theories suggesting that routine social actions, such as gossip, gestures, and grunts, indicate that people naturally strive to formulate identities ["Erving Goffman's Obituary," 1982]. Goffman refers to people's adaptive behavior as the various social roles and dramas in which we engage as social beings. From this approach, also referred the *dramaturgical approach*, we can extract several terms that could be useful for our own analysis of human interaction through their network devices [Goffman, 1959]. It is important to note, as highlighted by Meyrowitz [Meyrowitz, 1985], that this role-playing activity is not the exceptional case of pretense by people who are dishonest or not in touch with their real selves. Rather, Goffman's role-playing is one that functions as a crucial interaction mechanism found in every society. Specifically for our VPM discussion, we can denote the following terminology by Goffman:

*Actor/Performer* - a person in a given role, performing the duties that are consistent with that role (such as "waiter", "teacher", "housewife");
*Audience* - the people for whom we perform our roles (the audience members are also actors to each other);
*Performance* - the activity "given off" by an actor for their audience;
*Setting / Region* – the physical layout or background where interaction occurs. *Persona* - a mask worn to project a particular image to an audience;
*Impression Management* - how a person manages their peers impressions of how we act in interactions;
*Identity Negotiation* - the process by which roles are established, which makes interaction possible.

Goffman's terms are mostly useful to us when we look at individuals and how they present themselves to their peers. Goffman also points out the limited boundaries of an interaction region in physical space as well as boundaries with respect to time. For our discussion of how a social system functions, and how a "social sphere" is defined within a boundary, we will mostly rely on the work in the area of "*Behavior Settings*", as defined and studied by by Roger Barker and studied by Barker and associates [Barker, 1978; Wicker, 1987; Wright & Barker, 1959]. Goffman himself refers to this work and parallels to his own definition of regions [Goffman, 1959, p. 106].

According to [Barker, 1978; Wicker, 1987; Wright & Barker, 1959], behavior settings are small-scale social systems, whose components include people and inanimate objects, and are bounded by time and space. Barker notes that "a behavior setting is a place where most of the inhabitants can satisfy a number of personal motives, where they can achieve multiple satisfactions. Furthermore, different people achieve different clusters of satisfactions in the same setting". [Barker, 1978, p. 219]. The behavior setting approach looks at the setting as an entity of its own, that is governed by a setting *program*, and uses it to affect people's behavior. The program of a behavior setting is a time ordered sequenced of person-environment interactions that lead to the orderly enactment of essential setting functions. Behavior settings are viewed as capable of "defending" themselves against threats to their program. Settings "are superordinate, self-regulating, dynamic entities, which manipulate the behavior of their human components toward an equilibrium state for the setting" [Wicker, 1987]. If an important component is absent, setting occupants will seek it out and bring it. Similarly, if a component interferes with the program, the setting will "work" to replace it. This notion of a superordinate entity that both defines an interaction space for its child entities, and also affects their behavior is an important concept of the VPM framework. In the VPM framework we extend this to structures of multiple hierarchies of parent and child entities.

To illustrate an example of a behavior setting, Barker uses a gift-shop scenario. The physical and temporal boundaries of the shop are the walls of the rooms it occupies and its operating hours. Its components include the employees and customers, as well as inanimate objects like shelves, cash registers, money, and the goods for sale. To demonstrate the lack of dependence in a particular person in the case of the store, one clerk in the store can be replaced by another without the operation of the gift shop being seriously impaired. In our context we can think of a file server hard-drive crash, which causes the other devices, autonomously or with a helping human hand, configure an alternative device with the server functionality.

Literature and formulated theories like behavior settings could help us define the models of network behavior, adjusted in real time to the users' behavior patterns, or simply inspired by the way humans operate in a self-assembling way. It is important to understand, or at least try to understand how humans perceive and interact with their environment and their peers, and as we have shown, there are many parallels that can be drawn to data networks.

Anita Blanchard introduces the concept of "virtual behavior setting", which uses the idea of behavior setting for explaining people's behavior in online communities [Blanchard, 2004]. A sense of place is emerging in these computer mediated communication (CMC) communities, which inspired the adaptation of Barker and Wicker's theories of face-to-face, place bound behavior settings.

Work such as that by Cliffort Nass and associates [For example, Nass & Steuer, 1993] deals with computer actors, but in the context of a human-computer interface. In Nass's work, the computer acts in a human setting and with "human interfaces" like voice.

The works and concepts described so far lead us to several interesting points with respect to our discussion:

First, we should note that all of the related social work, including virtual behavior setting [Blanchard, 2004], deals with human actors and their interaction with each other. In virtual behavior settings, the locale is virtual or digital, but the actors are still human, with all the psychological and cognitive aspects that are an inseparable part of them. In our case, we want to create a framework that allows digital non-human entities to participate in pseudo-social interaction. Some terms, like "actors", "roles", "audiences", etc. are borrowed from the human domain, yet we will have to define a digital model for them. Other terms, like "setting program", originally borrowed from the world of computers, return to their alma mater. Nass's work is different since the machine actors are part of a human social situation, whereas our interaction space is purely digital, and is done through network protocols and related behavior.

A noteworthy point made by Blanchard is that a participant engaged in CMC is actually part of two behavior settings – the CMC's virtual behavior setting and the FtF behavior setting. This aspect is embodied in the examples we discussed earlier, related to the effects of mobile privatism and cocooning (see section 2.3). In this discussion, however, we also care about how being part multiple interaction arenas affects behavior in each of them. The fact that the user is physically at a home or work setting might affect their behavior in the CMC setting, and their participation at a CMC setting may affect their physical behavior (e.g. stopping a physical conversation to interact in the CMC domain). In addition, we are very much interested in the mechanisms that help a person *manage* all of the concurrent settings that he or she is part of, and make the decisions whether to turn different settings 'on' or 'off'. The VPM framework incorporates mechanisms that are meant to model this management of the digital milieu.

A point that Blanchard does not explicitly discuss but emerges from her work, is the notion that using CMC mediums, *a person may be active in more than one virtual behavior setting concurrently*. For example, being logged on to Second Life, a work related web discussion forum, an internet relay chat (IRC) channel and an Instant messenger session with a remote friend, all at the same time. Another example is a user opening different MySpace accounts under different aliases, and participating in a completely different virtual behavior setting for each of them. Scenarios involving multiple-behavior-setting such as these are very common

in today's online interactions using a personal computer. However, they are not common at all in traditional human FtF communication, where a single behavior setting emerges from the collection of people and objects in one's vicinity [Barker, 1978; Goffman, 1959]. This is also referred to by Meirowitz, as he talks about a single performance that a person "tunes" himself to according to the aggregation of the current audiences in the vicinity [Meyrowitz, 1985]. Within a certain scope, a single network device may present multiple different identities in ways that make it hard if not impossible to realize that a single device is behind them. Some might consider this a vulnerability of the network architecture, as it might allow offensive or fraudulent behavior without accountability, but under other circumstances this gives us interesting new capabilities for realizing social interactions. We could devise mechanisms that allow a single wireless device in the FtF networking space to be part of two or more digital behavior settings concurrently, without allowing parties in either setting to be aware of information they are not eligible for. Aside from allowing the personas of both Clark Kent AND Superman to be present concurrently in the same ad-hoc network without risk of exposure, the importance of such capabilities is discussed in detail in section 3.4.2, and our proposed implementation, which is the core of the VPM architecture, is presented in section 4.2.

Figure 4.1 depicts the different combinations of human and digital settings and actors, and the different theories and works dealing with each. We should note that although the VPM framework has machine actors interacting in a digital realm, their social situation is very much connected to the physical world. These devices move about the physical world. They adjust their digital roles and performances as they do so, according to the physical world's context.

| Setting / Actors | Real world | Electronic |
|---|---|---|
| Human | Barker, Wicker (Behavior Setting) | Blanchard (Virtual Behavior Setting) |
| Machine | Nass (Machine actors in human realm) | **VPMs (Digital Behavior Setting)** |

**Figure 4.1 – Different types of behavior settings and actors.**

### 4.1.2 Stream of Behavior

Another key concept described by Baker is the *stream of behavior* [Barker, 1978, p. 49], which discusses how people's behavior is a continuous stream, very different than their previous experience with laboratory studies. In the latter, both behavior and environment are arranged in discrete segments (interviews, experiments, tests, etc.). He also defines the concepts of *behavior units* and *behavior tesserae*. Behavior units consist of the inherent segments of the continuous stream of behavior mentioned earlier, whose boundaries units occur at those points of the steam where changes occur independently of the actions of the investigator. Behavior tesserea are fragments of behavior that are created or selected by the investigator when he functions as an operator, arranging conditions in accordance with his scientific aims.

These observations emphasize the importance of being able to deal with the dynamic day that the mobile users go through. When looking into networking literature, it seems that a large part of the discussions and performance experiments deals with two main scenarios: relatively steady states, or "step" function scenarios. In the former, there are given initial

conditions, and the network performance is observed under them (e.g. congested network vs. lightly loaded network). In "step" function experiments, there is a given initial state, and then a radical change in the condition, after which the transition function is observed and analyzed. The idea of the stream of behavior might hint that it would be favorable to implement mechanisms that adjust to the current state of the network and the user and have some memory, assuming that life does not change radically in an instant (though it sometimes might). Behavior units and tesserea could be paralleled in the networking world to network timers and events that occur according to preconfigured timers vs. those that occur based on network events or other sensed events. Behavior tesserea could be paralleled to lab tests and predefined experiments, or to network timers and events that occur according to preconfigured timers, as opposed to algorithms that execute events based on network activity parameters or other sensed events. In addition, many computational social science experiments that make use of network parameters or sensors use periodical scanning of the environment, mostly for conserving battery life. For example, the reality mining project [Eagle & Pentland, 2006] used Bluetooth scans at 5 minute intervals. As Barker points out, those intervals impose skews on the beginning and end times of recorded events, or may under-sample reality during periods of very dynamic changes. We can use Barker's observations to justify development of network events and protocols that are better aware of the natural behavior units of the human users, which will enable both more accurate performance to the users, and more accurate measurements of human behavior. For example, a desirable service discovery protocol might use network activity or human activity sensed by external modalities in order to decide how often to perform radio scans for devices in range. This might conserve battery life when the setting remains static, and also provide faster response in highly dynamic situations. It might make better sense to adjust our devices to the time scales of their human users rather than the processor's clock speeds. In form with the generalized goals of Social Area Networks that we describe in the next chapter, this approach will also help us shed some of the default parameter values and replace them with dynamic, personalized parameter configuration for the user.

### 4.1.3 Categories, Groupings, and Audiences

In "Women, Fire, and Dangerous Things" [Lakoff, 1987] George Lakoff touches on the centrality of categories in people's lives and cognitive processes. According to Lakoff, categorization is the most basic element of our thought perception, action, and speech (p.5). "Most categorization is automatic and unconscious" [Lakoff, 1987](p.6). In "No Sense of Place", Joshua Meyrowitz's discussion of audiences and audience-specific performances and interaction directly reflects this idea [Meyrowitz, 1985]. The book opens with a very good example of how people tell the same story in different ways to different audiences, in his recollection of his own trip to Europe as a college student (p.1). One of the interesting things about Meyrowitz's work is that he talks about social situations, or behavior program, which is dependent and set by social setting rather than physical place. This is very relevant to the VPM framework, since most of the information a device has is its ability to sense its social setting – its currently active network peers.

Fernanda Viégas' MS Thesis, "Collections" [Viégas, 2000] deals with many overlapping aspects of human interaction. Especially there is an overlap with relation to categorization and grouping of different audiences. Viégas' work deals with categorization and

management mechanism for large collections of personal content and personal information through online applications and the World Wide Web. Collections deals mostly with interfaces for managing content sharing with different peers and groups, but the more important distinction is that Collections deals with audiences, with a user's performance towards the outside, while in the VPM framework we look *inside*, into the users – how their multiple identities, affinities, and personas are arranged and managed. In addition, Collections looks at a single actor vs. a diverse audience. The system is asynchronous – there is a strict separation between the offline process of organizing the collection and configuring its presentation and the actual peer that connects to view the content at a later time. In VPMs, even though configuration persists and can be initialized in advance, much of the interaction occurs synchronously, in real time. In VPMs, we look at the emergent social system of many actors and many audiences intertwined simultaneously. A common aspect to both systems is in how they encourage the users to "articulate some of the various categories of social relationships they are a part of. Audience circles such as 'friends', 'family', 'coworkers', and 'best friends' make social bonds explicit." [Viégas, 2000]. For the social research aspect of this work that deals learning about human social relationships, this aspect of the system is a great way to learn information that was not otherwise available. Instead of relying solely on statistical analysis and inference of social relationships, the users themselves enter this information (the level of accuracy of this information is a matter of a separate discussion).

### 4.1.4 Private and Public

Goffman talks about the back stage and the front stage of an interaction region [Goffman, 1959], but a similar thing occurs within our minds as well. We have a set of behaviors notions, and information that we expose and make explicit to our peers, and there is still information that remains in the privacy of our minds. Japanese culture terms these two realms of private and public as "*Honne*" and "*Tatemae*". *Honne* refers to a person's true feelings and desires, which might be contrary to what society expects or requires. They are often kept hidden. *Tatame* refers to a person's façade. It is the behavior and opinions that one displays in public [Doi, 1973]. As we implement our technical solution, we need to keep these issues in mind. We need to offer tools that allow the user to separate between what is private and what is public – for example allowing the user to give groups a private names or generate groups that fit some personal category but do not exist outside of his device. The second thing to be aware of, and perhaps even more important is this: Part of what the VPM framework aims to do, is extend the user's Honne into the device. As we do so, it is imperative to keep it private to the user and not expose information that they would like to keep private through alternate channels. For example – exposing to a peer the user's true and negative feelings toward them. This is very relevant to the interaction design aspects, and we touched on this in our discussion of interaction design for the hybrid interaction space in section 3.5

## *4.2 VPM Framework: Basic Concepts and Requirements*

At this point, now that we reviewed relevant social and psychological literature and before we get too technical, we introduce some of the key elements of the VPM framework with the aid of a short narrative. Rather than doing yet another Alice and Bob story, our scenario

is inspired by "*The Princess Bride: S. Morgenstern's Classic Tale of True Love and High Adventure. the 'Good Parts' Version, Abridged.*", by William Goldman [Goldman, 1973]. This tale combines intrigue, stereotypical categorization, multiple audiences, persons holding multiple identities, relationship tensions, various behavioral settings, and other key features that make it a good candidate to be casted onto our own discussion[9]. Evidently, this also fits well with Goffman's dramatulorgical approach.

## 4.2.1 Network Structure, Network Knowledge

### 4.2.1.1 Sample Narrative: Part 1 - Meet the cast

Figure 4.2 depicts the actors and personas of our example, as well as some notations that will be used later on. So, without further a due, let us meet our cast:

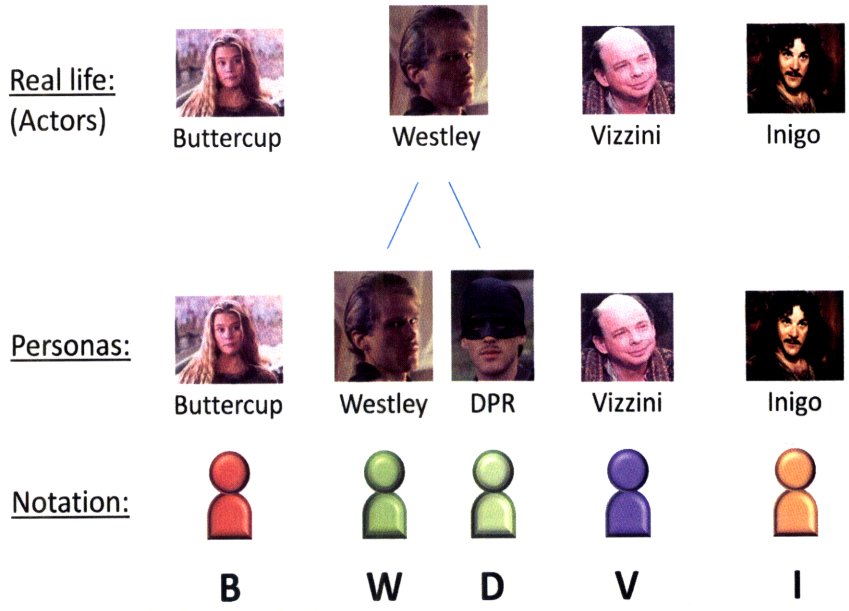



Figure 4.2 - The cast of our example narrative.

- ***Buttercup (B)*** (yes, apparently that is a name) is a princess by birth. She likes walks along the beach, lovely sunsets, and cute puppies. At the moment, Buttercup's social network looks something like this:
    - Westly, her fiancé, who she loves.

[9] For the record, if this section ever turns into a published paper, it would be titled: ***My Double Life as a Pirate and Stable Boy***: *The very-much-unabridged fictitious tale of Westly and his associates in the land of Virtual Private Milieus based on W. Goodman's abridged "Good Parts" version of S. Morgenstern's Classic Tale of True Love and High Adventure.*

- Dread Pirate Roberts (DPR), who she dislikes because she believes DPR hurt her fiancé, Westley, but still wants to talk to him in order to find out about Westley's whereabouts.
- Vizzini, who she does not particularly like since he is trying to kidnap her for ransom.

If Buttercup had an instant messenger account or was part of an online social networking service, she would probably have all three in her contact list. She would expose her presence to Westly and the Dread Pirate Roberts (because she wants to interrogate him about the whereabouts of Westley), but she would set herself to be "invisible" to Vizzini, since she does not want to expose herself to him

- ***Vizzini (V)*** is a smarter than average criminal. He likes riddles and really loves money. That's why he wants to kidnap Buttercup and ask her father for ransom. Vizzini's social network looks something like this:

  - Dread Pirate Roberts – a fellow riddle lover and nemesis.
  - Buttercup – a kidnapping target.

  If Vizzini had an instant messenger account or was part of an online social networking service, he would probably have both of them in his contact list. He would want to be notified when either is around. Having the large ego that the has, let's assume he would want to announce himself to both of them.

- ***Westley*** is Buttercup's fiancé. However, Westley the person actually holds two virtual identities, or personas: The first is his true self – ***Westley (W)***. The second identity is ***Dread Pirate Roberts*** (***DPR*** or ***D*** for short). The Westley persona has a relationship with buttercup only.

  While in the DPR persona, Westley chooses to expose his riddle loving side. It also allows him to act in ways which he would not feel comfortable doing as Westley. The DPR persona has two contacts in its portfolio of sociability:

  - Vizzini – a fellow riddle lover and nemesis.
  - Buttercup - DPR's connection with buttercup is very different than the connection his Westley persona has with her. Buttercup is not aware that both personas belong to the same person/actor.

  In traditional human FtF interactions, as studied by Goffman and Barker, Westley would only be able to present a single persona at a given time, but in the digital realm it would be possible to present both concurrently. The VPM framework allows the user to manage these personas. Neither Buttercup nor Vizzini are aware that both identities serve the same person, and Westley would like to keep it this way.

- ***Inigo (I)***. His name is Inigo Montoya. He is on a quest to avenge his father's death, and believes that his father died at the hand of Dread Pirate Roberts. We shall leave Inigo out of the story at this point, but he will come into play a bit later in our discussion.

With this basic story overture, we can already start discussing some of the basic terminology of our approach. We start by looking at different aspects of the internal and external

knowledge that exists about the network and the different connections between our actors and personas.

### 4.2.1.2 Omniscient network/social view

We'll define the ***omniscient view*** of a network or a social structure is defined as the full range of information that is part of the network. Figure 4.3 shows the different connections between the networked personas, which is the full view, or *omniscient view*, of our current social network. Note that the connection between B and V is unidirectional, representing the fact that B would not want to expose her presence or any other information to V, whereas V *would* reveal his presence to B.

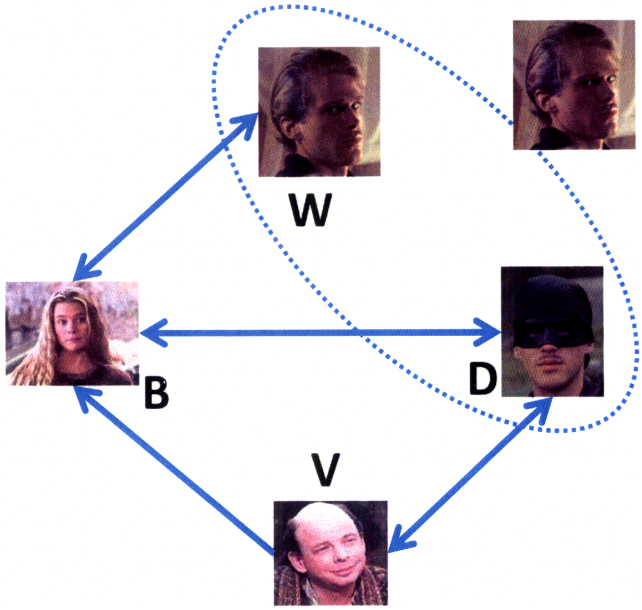



Figure 4.3 - Cumulative network overview.

We should ask ourselves – in real life, who has this full network view? Who actually needs it? As researchers we might want to collect and analyze it, and had we been using a centralized online social networking tool, such as Facebook [Facebook], that information would be fully available to the organization providing that service. But is this information really needed for providing the communication services desired by the users?

The omniscient view approach has potential for multiple caveats, which we want to sway away from:

- Scalability – It gets very hard to manage as the number of nodes increases, especially when there are different types and weights for the connections, and they are not symmetrical.
- Potential privacy risk – all information is centered at one or more points. This opens the door to misuse, either by having the information stolen or leak out by mistake, or even misused by the official keeper of the data (say for advertisement purposes,

such as the uproar that followed the initial launch of Facebook's 'Beacon' system[10]). In addition, some of the connections between individuals might be very discrete and much more sensitive than others. For example in the area of romance.

- The omniscient data could lead to unnecessary requirements and dependencies:
    - Keeping the information up to date for all nodes – dealing with contradicting information, errors, propagating changes to network structure.
    - If some of the information is kept remotely, such as on servers, this requires a connection to the remote location, and if it is not available, the service may be down. For example – Loopt's service [Loopt] will not work if there is no live Internet connection.
    - If information is cached on the end devices, much more space would be needed for saving the information, and probably not all of that information would be actually utilized by the user.

Only in our capacity as social researchers will *we* ever have, or *attempt* to have, the omniscient view. We will design our distributed discovery and communications algorithms as if we do not have access to the full view of the network.

#### 4.2.1.3 Individual network/social view

Our approach is to look at the network from an individual's perspective. In contrast to the omniscient view of Figure 4.3, Figure 4.4 examines the individual network picture of the four personas. As we mentioned earlier, this makes sense from different aspects:

- This is more similar to how humans operate with their real-world networks and relationships.
- This allows for scalability in management and control– each individual maintains their own network and network view, incorporated into their own devices and maintained by them.
- This allows for flexibility in the depiction of the network. Similar to real life, any link between two people can be viewed and tagged differently by each of them, and also by others who are aware of this connection (by learning about it from peers or on their own, as "onlookers"). For example, the asymmetric connection between B and V, which in our initial case is binary, but may also be more granular, for example by adding a trust parameter between 1-100, which affects different behaviors and performances between the two personas.
- Designing our mechanisms to work with te limited personal view does not limit them from having enhanced performance in cases where more information is available.

For the most part, we are going to assume that the users, their devices, and our algorithms do not have the full picture of the social system or the network structures (of people or their devices).

[10] See [Aspan, 2008]

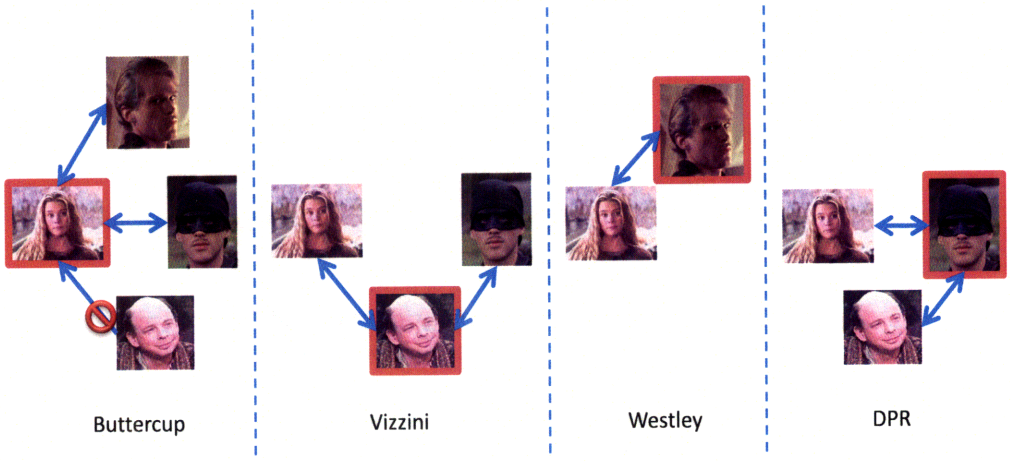



Figure 4.4 - Individual Network View. "Owner" of the perspective view is highlighted.

### 4.2.1.4 Learning additional network/social information

Just like a person is able to infer a relationship between two of his peers by seeing them react and communicate with each other, as our network devices roam the world they will also be able to use their radio receivers or other "senses" to notice their peers communicating with each other through a shared transmission medium, and possibly infer a relationship between these devices or their owners.

Just like in the real world of FtF human interactions, two persons may choose not to publically expose their relationship or acquaintanceship when others are nearby. In the networking world, two parties may choose not to expose their relationship using existing privacy technologies, like TOR [Cerf et al., 2007], [Zimmermann, 1995], or tools still in development, like Tryst and Shroud [Greenstein et al., 2008; Pang, Greenstein, McCoy et al., 2007], Smoke Screen [Cox et al., 2007], or Private Set Intersection [M. J. Freedman et al., 2004; Hazay & Lindell, 2008; Kissner & Song, 2005].

Another way to learn about peer's connection with each other, is by sharing such information with trusted parties. For example, all of a single person's devices will share and complement their network view to the fullest extent of their abilities. Devices of people with a highly trusted connection may also share all or part of their network structures, with varying levels of detail. For example, devices owned by family members or close friends may share with each other some subset of their network and relationship. For example in the UMI architecture relationship information is shared through a gossip algorithm, though in UMI ALL information is shared [Ford et al., 2006], and that might be too much for the VPM approach.

### 4.2.1.5 Resulting VPM requirements

This discussion leads us to the following requirements of our VPM framework.

- Individual network picture as default assumption
  - We want to look at social OR communications network from the personal perspective, and personal context. We are the center of our own network, and manage our own portfolio of sociability.

- Allow different types of relationships / network links
    - Save this information
    - Allow this information to inform behavior (e.g. remain invisible to some peers by choice).
- Support multiple identities/personas per device, orthogonal to one another.
    - Need to support logical separation between identities and interactions to preserve privacy among them.

### 4.2.2 Shared Settings

#### 4.2.2.1 Example Story: Part 2 – Adding some network devices.

At this point we are going to assume that each of our human actors carries at least one mobile device with them, capable of device-to-device communications. For example, we will give each of them a WLAN (802.11) capable mobile phone, which runs the VPM framework and is configured with that individual's social parameters and settings. These devices are able to work in the WLAN's ad-hoc mode, meaning they have the capability to communicate directly with each other in a device-to-device mode. However, in order to be able to actually communicate, the devices must first be in range of each other, and also have a similar radio interface configuration. For example, they would have to operate on the same radio frequency channel (out of possible 13 channels in 802.11b or g standards [IEEE 802.11].

In networking this is simply referred to s being part of a *shared transmission medium*, or simply a *shared medium*, which is a basic term in data networking. A shared medium refers to a shared transmission medium where multiple connected devices have access to the medium and have to share the usage of the medium. Access and usage of the shared medium is managed by Media Access Control (MAC) mechanisms, which are part of layer 2 of the OSI model [H. Zimmerman, 1980]. Examples of this concept are all devices that are associated to a WiFi access point, devices that are part of a passive optical network (PON) tree (IEEE802.3ah), or devices connected to an Ethernet bus. A shared medium could also be emulated using overlay network platforms, like peer to peer file sharing or RON [Andersen et al., 2001].

We will now assume that all three actors (or 4 personas) and their mobile devices happen to be in the same vicinity in terms of the wireless device's range, as depicted in Figure 4.5. They might even be part of the same shared transmission medium (i.e. same wireless channel and ad-hoc network), yet this still does not guarantee they will be able to detect and communicate with each other. A technical shared medium is not enough to guarantee shared communication between peer nodes. They would have to support the same communication protocols like addressing schemes, messaging definitions, transmission handshake, or any other encapsulated protocols like encryption schemes. Running the VPM framework would guarantee a matching of the general communication protocols, but different entities may choose not to automatically participate in the communication and expose their presence, for example. In our narrative, we already defined Buttercup's desire not to expose her presence to Vizzini. This means that her device may learn of Vizzini's presence in the network vicinity, but not expose its own presence.

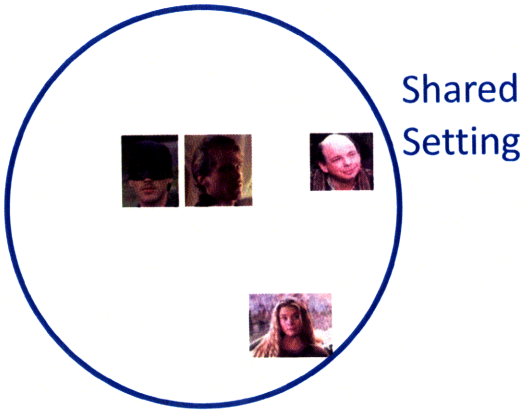



Figure 4.5 - Shared setting example, omniscient view.

### 4.2.2.2 Basic Shared Setting

As we discuss in section 4.1.2, today's digital media allow us to participate in multiple digital behavior settings simultaneously. We might say that this allows us to re-erect the walls that Meyrowitz noted to have been demolished by new media of the 1960s [Meyrowitz, 1985]. Today's technology allows us to establish virtual walls, and choose who is included as well as who is excluded. Using regular Internet applications like an instant messaging or an IRC chat server, Buttercup would have been able to create multiple private chat rooms with her friends thus creating "walled gardens" for invited members only. Her instant messenger's "buddy list" is a sort of shared arena of its own, or a set of shared arenas for pairs of users – her and any one of her "buddies" on the list. If she desires, she could mark herself as "invisible" to Vizzini while still being able to see that he is online. From the networking perspective, this is more than just a shared medium, but a logical segregation imposed over a shared medium. From the behavioral sciences perspective, these arenas are still not behavioral settings on their own (although they might spawn behavioral settings). In the spirit of convergence and generality, we will coin the term "***shared setting***" to treat all of these types of interaction arenas. A shared transmission medium is also a shared setting for interaction between devices. We give a more accurate definition of shared settings and related terms in section 4.3.

Devices that are part of a shared medium and execute a protocol stack that allows them to openly discover and communicate with each other would be considered to be part of a shared setting. For example, devices on the same ad-hoc network running the TCP/IP stack and also the Zeroconf [Williams, 2002] framework will be able to discover and communicate with each other, at the network level. Note that by "network level" we refer to the set of messages that Zeroconf allows, like device and service discovery messages. For enabling user communications, all peers desiring to communicate would have to run user applications that integrate with the stack and implement those communication protocols.

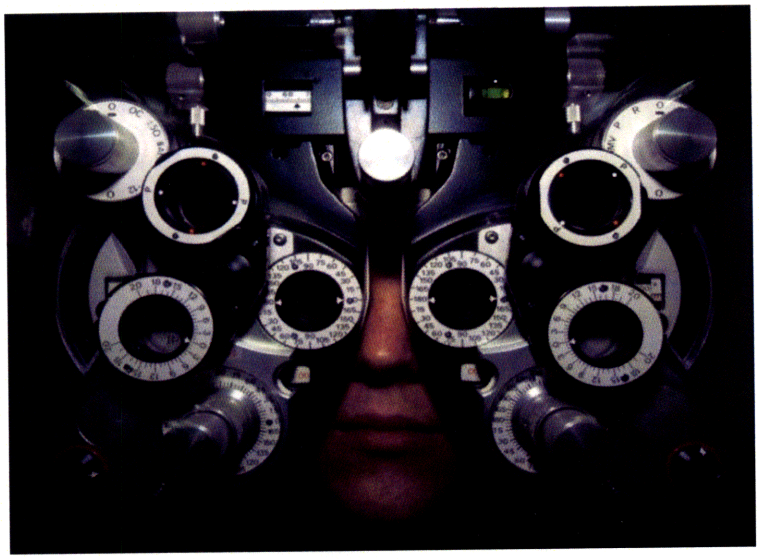


Figure 4.6 - Optometrist phoropter tool.
(Source: http://en.wikipedia.org/wiki/Image:Geraet_beim_Optiker.jpg)

However, as we defined our system, in line with our analysis of the *hybrid interaction space* in chapter 3, we do not want all personas to "see" each other in the shared medium. We also want a system that would prevent any onlooker from realizing that W and D are actually two personas emanating from the same person and device. The desired outcome of our example where the 4 personas are present in the same shared setting is that each persona will only expose itself to the peer personas that it wants to be seen by, and as a result only be able to discover those peers that allow themselves to be discovered by that persona. When we think about it, this can be paralleled to each of the users looking at the world through filtered shades, and also projecting their own image through another selective filter. This would be something like the metaphorical equivalent of the phoropter depicted in Figure 4.6.

The resulting view of the world is unique or personalized to each entity or persona, and is a function of their personal filter function as well as their peers'. Figure 4.7 depicts the result of this exercise for the filter rules that we have defined in our story. Vizzini will only see the presence of DPR, the single peer that he has a connection to *and* also agreed to be seen by Vizzini. Westley and DPR are two personas of the same user, so he is able to see the aggregation of the filters of both personas. Buttercup sees three peers in her vicinity, since she has no way to know that W and DPR emanate from the digital aura of the same person.

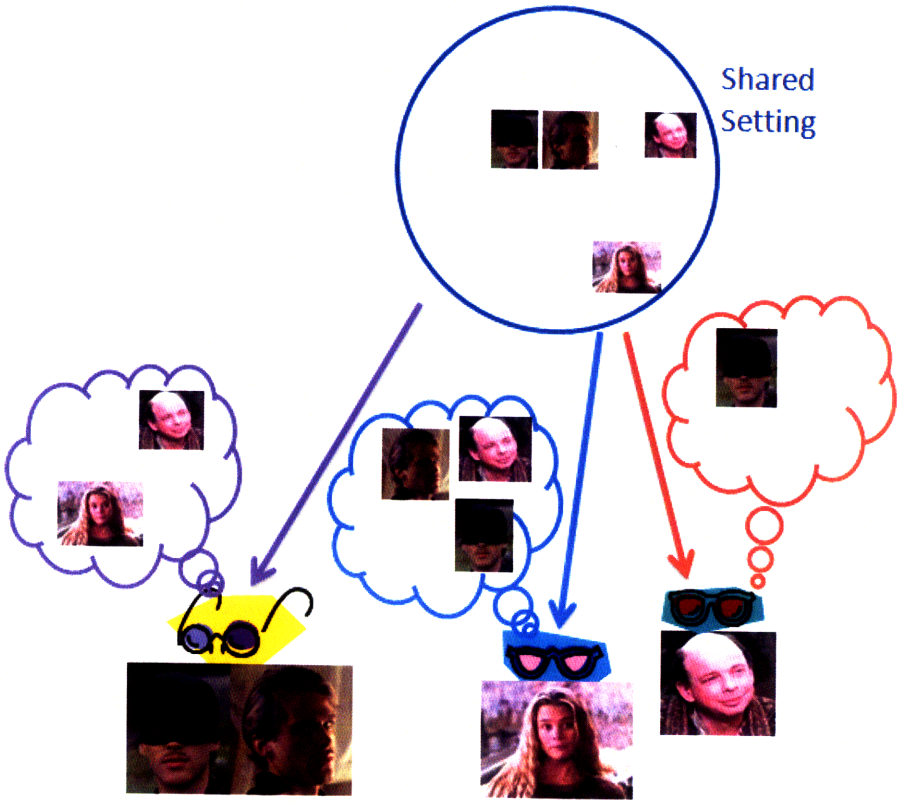



Figure 4.7 – Filter metaphor for selective peer discovery and interaction.

### 4.2.2.3 Sub-Settings

There is another requirement to make this mechanism work – The users need to have the option to create "private" settings of interaction that would enable them to interact with desired peers without being interfered or eavesdropped. There is no point for B to hide her presence from V, if V could overhear B talking to W within the common shared setting and infer both her presence AND ther attempt to hid from him. This issue is discusses in more detail in chapter 3, where we touch on the sensitive signaling aspects of the hybrid interaction space. Using our terminology from above, we would like to divide the shared setting created by the shared transmission medium to walled *sub-settings*, just like in the case of private chat rooms or the instant messenger's buddy list. Each of these sub settings is a shared setting of its own, limited to a subset of the entities in the shared medium. If desired by the users, we want the ability for these sub-settings to be split down even further, to yet smaller groupings and hierarchies. We also want to give users the ability to decide whether to make a setting they "own" private or public for onlookers or lurkers, however we will not go into the detailed nuances of the shared setting concept in this part of our discussion.

There are many different combinations of potential shared setting "filters" that we could impose over the shared medium. In fact, for 4 personas, or entities in the network, there are a total of $2^4 - 4 - 1 = 11$ potential shared settings, and for n entities there are $2^N - N - 1$ potential settings. In a way, this is a realization of Reed's law for value created by a group-forming network [Reed, 2001].

We do expect the actual number of active settings that would be used to be much sparser, but it is good to keep the potential in mind. Initially there might seem to be no logical reason for implementing a shared setting between W and D since they belong to the same person, but we can definitely think of a scenario where Westley the person has two devices, each configured to a different persona, and he would like to create a private shared setting for the two devices to connect and exchange files between them or share their network and relationship knowledge. We can also find use for a setting configured to a single persona, say Vizzini, and we discuss this later in this chapter, where we talk about users having multiple devices configured with the same personal identity.

### 4.2.2.4 Nested Sub-Settings

Figure 4.8 depicts two possible configurations of shared settings that fit our narrative. Both include a top shared setting, that includes every participant - this is in fact the setting defined by the shared transmission medium. We define a ***top shared setting***, or ***top-setting*** for short, exactly as that – the highest level of shared setting, that includes the largest set of possible entities. This highest level will usually be defined along the physical limitations of the shared networking medium. Aside from that top setting, we have three private settings for three pairs of personas – {V, D}, {B, D}, and {B, W}. This would allow each of those pairs to conduct their own private interaction. Since there might be situations where B would like to send a message to all of her connections aside from V, we defined a shared setting that includes the three entities B, D and W. We can define their setting either by the entities included in it {B, D, W}, or by who is *excluded*, from it, V in our example (so we could also name it "NotV", or "!V").

The difference between the two configuration examples depicted in Figure 4.8 is the relationship between the different shared settings and the way they are nested within each other. In the left figure, all three pair settings and the {!V} setting are next to each other, nested under the hierarchy of the top setting. In the right figure, the two settings of {B, D} and {B, W} are nested under a new shared setting {B, D, W}, which includes all of their participating entities.

The difference between these two configurations relates both to aspects of privacy and aspects of perception. Walking down a street, I can see houses, count their number, and might even see who goes in and out of them. However, I cannot see into the internal division of rooms within a house. Once I am allowed into a house, I would be able to roam it and see at least the number of rooms that are in the main corridor, as well as the individuals that might walk in and out of them. As we decide on the topology of a desired shared setting, we might consider towards who we want to aim our privacy concerns. Sometimes we want to hide our private interactions from onlookers on the street or in the top setting, which would make it better to hide our new private room within the walls of an existing house number. Other times we might want to hide our private collusions from members of our own metaphorical house, in which case we might choose to leave the existing walls and erect a new house in a remote part of town. As we define interfaces for users to set and manage their shared settings, we must make clear to them what kind of privacy each configuration gives them (see chapter 5).

We would also note that Figure 4.7 depicts the omniscient view, and as we mentioned earlier, none of the entities will be privileged with this complete picture. We will discuss this more soon, when we deal with security and privacy aspects of VPM shared settings.

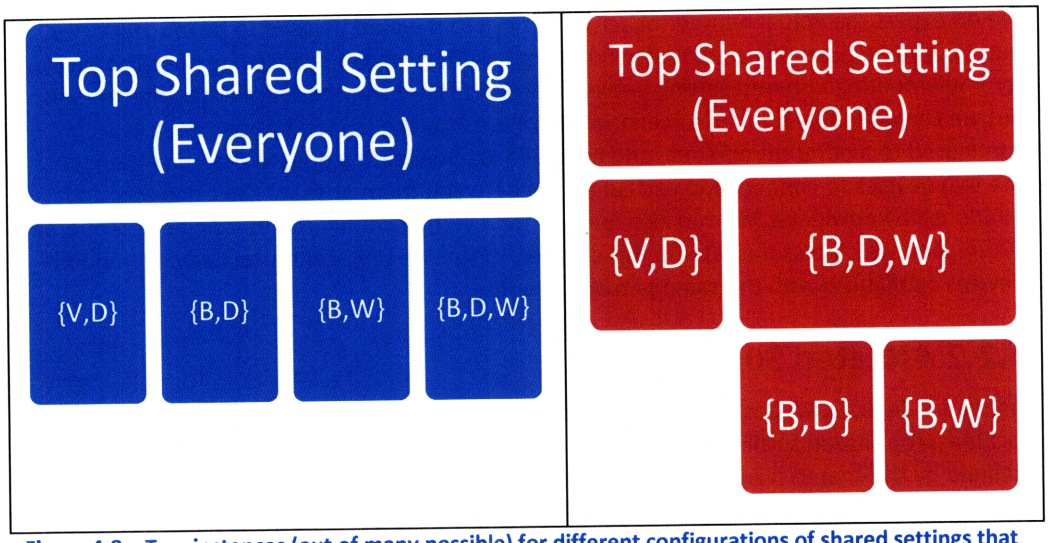



**Figure 4.8 – Two instances (out of many possible) for different configurations of shared settings that could be implemented with the four digital personas of our story.**

### 4.2.2.5 Discussion: Interaction Arenas and Shared Settings

In either the physical or digital world, for any interaction between entities to occur we need a shared arena and medium to connect the different entities and enable communication - allowing at least one party to emanate a signal, and another party to receive it. In the traditional human realm such arenas are created by simply having persons co-located in a distance that enables one person's senses (like hearing, vision, or touch) to receive signals emanating from a peer (like sound and visual or physical actions). This enables what we call a face to face (FtF) interaction, which in turn enables the creation of Goffman's *regions* or *stages* [Goffman, 1959] or Barker's *behavior settings* [Barker, 1978].

Digital media allows virtual spaces of interaction, like Blancard's naturally occurring social spaces that act as the arena for her virtual behavior settings [Blanchard, 2004]. The digital world allows countless others shared arenas of interaction that are so pervasive and common these days that they might be taken for granted. Even email could be considered as an asynchronous type of a shared setting, where creators of messages choose which of their acquaintances to include in the setting. The setting lives as long as the message recipients keep participating in the resulting thread using the "reply to all" option. Any selective reply, to a subset of the original recipients or to new recipients creates a new shared setting.

We could describe the nesting and hierarchies of sub-settings with McLuhan's terms, who describes how the content of each medium is a medium on its own [McLuhan, 1964]. Here it is very much obvious – within the scope of the VPM domain, each shared setting is a form of medium, which can recursively house additional media of the shared setting type.

### 4.2.2.6 Resulting VPM requirements

This sub section presented a number of key ideas that are in the foundation of the VPM framework. Here is the summary of the key requirements that we want to add to our distributed interaction system:

- Create a common framework that would allow networked devices to rendezvous as part of a shared transmission medium and act as a top setting.
- We want to define a software implementation of the shared setting concept, nested within the top setting, and allow their *distributed* creation and management by users. We need to implement shared-setting management functionality - basic actions such as creating, deleting, merging, structuring in hierarchies, inviting or rejecting, etc.
- Need for mechanisms that perform the actual discovery process, i.e. implement the function that for each user aggregates the different filters to create the full picture of the network at any given moment.
- Need for security and privacy mechanisms to implement the user's desires and view of the network.

## 4.2.3 Groups and Affinities

### 4.2.3.1 Motivation: Groups and Affinities

The previous section discussed shared settings from the perspective of individual communication. It also mentioned the idea of creating sub-settings that include different subsets of the participants. Expanding on this notion, we introduce the idea of group identities that serve as the basis for a shared setting. Rather than naming a setting according to the set of entities that are entitled to be part of it, we can define the construct of a *group*, (and perhaps also an *affinity*, to denote a lesser commitment than a group). Using groups makes sense for several reasons, the first of which is that, as we saw in the theoretical background section, groups and categories are engrained in the way people categorize information. It seems useful to extend this cognitive aspect to the digital realm as well.

Using groups simplifies the amount of information that has to be saved and processed by the network device, as well as the user. By changing the group's settings, for example the trust that the user has with its members, the change could be applied to all members of the group. However, groups are much more than simply replacing a set of entities with an alias. The members of the group can be explicitly enumerated, but they do not have to be. A group identifier, for example a unique key, could be used as a rendezvous point for activating a shared setting between people who have never met before. This could be useful for serendipitous introductions, or for dealing with groups whose membership is very dynamic.

### 4.2.3.2 Example Story: Part 3 – Adding Groups

Now it is time to add some groups to our story, in order to understand what they mean in our framework and get some notion of the various types of groups that are possible.

- Let us assume that Vizzinni has expanded his villainous enterprise and hired two employees – Inigo, and Fezzik (who will not make an appearance in our rendition of the tale). He has created a private group for them, called "Vizzini's Trio". This will allow them to have discrete communication with each other when they are in close vicinity, hopefully without being detected or eavesdropped upon. It also means that no one else can add themselves to the group or invite others without being the group's administrator.

- Vizini is also part of a group called "Sicilians", another private and secretive group for all Sicilian Crime-Lords (or at least those that see themselves as this).
- Vizinni part of yet another group, which DPR is also a member of, for riddle lovers. This group allows riddle lovers to discover each-other when they come within proximity of each-other, even if they had not met before – simply by having the Riddle Lover's group unique private/public key pair (other secure identification methods are possible, PKI was selected just as an example). The group is defined as "semi-public". According to the arbitrary definition of our example, we define this type of group as a group that is exposed to everyone and anyone can add themselves to it or invite others to it. However, in order to see the actual identity of group members in the vicinity, as well as communicate within the group's shared setting, one has to register as part of the group. This means that one would have to expose one's self, at least partially, in order to be exposed to "inside" information. This idea is in line with the symmetry principle that we mention in section 3.5.
- Buttercup is part of a public group called "Damsels in Distress", which is known to be globally used by damsels who find themselves in distressful situations[11] and is used as the digital parallel of yelling for help. It is defined as a public group, which means that one does not have to be registered to it in order to see who in the vicinity is part of it and see content that is sent through it. However, because it is set as a defined group, users can use this to change the way they treat it - for example, block all messages originating from it or highlight it when it is discovered. It is also useful for the members of the group, as it allows them to choose when they want to expose this identity of theirs, and under what context. When there is no "distress", there is no need to even activate this setting or advertise one's presence in it.
- Finally, all of our actors, by default, are part of the group called "Everyone", which is parallel to the scope of the top shared setting. This setting allows broadcasting messages to everyone in the vicinity, but as we hinted above, it does not mean that everyone in the vicinity will choose to accept those messages and might filter them out, according to their individual preferences.

Figure 4.9 shows the summary of the groups that each actor is a member of. Figure 4.10 shows the individual shared settings that result when all of our actors/personas are present under the same shared medium. Each block represents a shared setting. A block defines with a peer entity's name means it is a private shared setting for both the individual and the peer. It is immediately apparent how each actor sees a different view of the shared space. Things are starting to get interesting, not to say complicated.

[11] Incidentally, none of our actors have chosen to be part of the gender-neutral version of this public group "I've Fallen and I Can't Get Up".[†]

[†] Which replaced the failed attempt of the group "Help! I Need Somebody. Help! Not Just Anybody" who's name kept getting truncated by the small screens of most mobile devices.

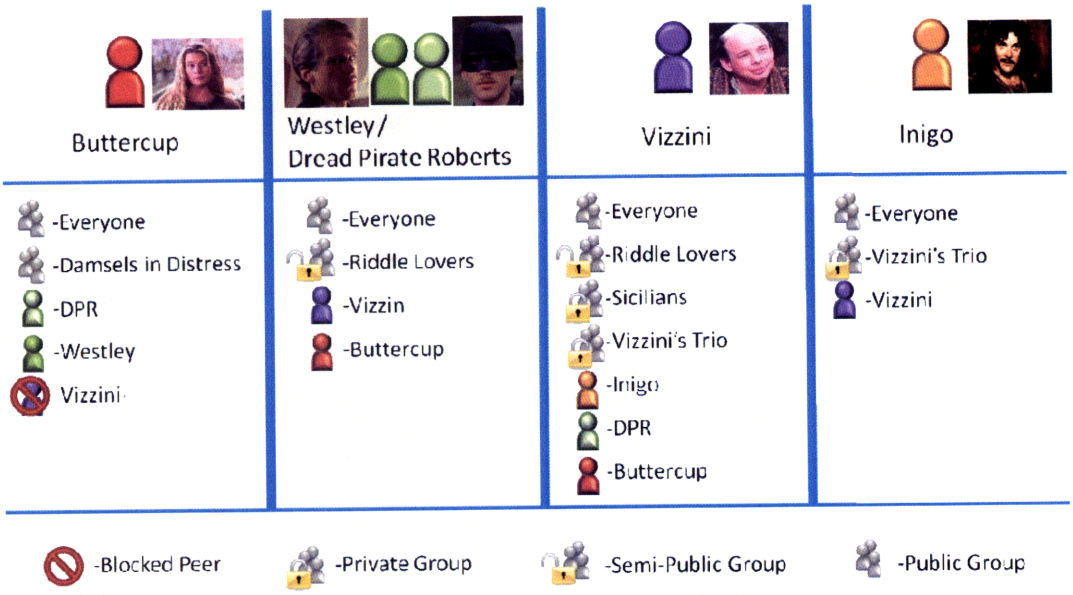



Figure 4.9 - VPM Groups and peers for all four actors.

We shall note some key aspects that arise from Figure 4.10:

- Sicilians group: Since Vizzini is the only member of the private Sicilian group, the shared setting is not active in our scenario. It takes at least two to tango.
- All members belong to the default group that represents the top setting.
- Vizinni's Trio: Only the members of this private group are aware of it (A potential attacker would see it as gibberish, or noise).
- Riddle Lovers: All actors see the presence of the Riddle Lover's group, but only the ones that have "registered" to it can see the individual members that are present.
- Vizzini and Inigo: As we defined our scenario, V and I only relate to each other through their business association and subsequent membership in the Vizinni's Trio group. They do not have each other in their "buddy list" as private individuals. Either one of them may choose to "turn off" their announcement of membership in the group, and become invisible to the other. For example – Inigo may choose not to be visible to his boss after work hours.


**Figure 4.10 - Resulting shared settings for each actor.**

- "The Buttercup Dilemma": The way we constructed our story leads us to an important case of signaling and interaction design that anyone dealing with building online and hybrid-space applications should at least be aware of. Buttercup wants to remain invisible to Vizzini. However she also wants to participate in a public group, which might expose her presence to *anyone* in the vicinity. There are

multiple possibilities to deal with this, which is why we left the identity of Buttercup as "???" in Vizzini and Inigo's view of the "Damsels in Distress" (DinD) group.

- Option 1: Buttercup or the system's designer did not care about this issue, or failed to deal with it. In this case B's regular persona would be exposed through that group to anyone in the vicinity who chose to view the group and its active members in the vicinity. This would expose buttercup's presence to V, and might lead to him realizing that even though he used to be a personal "friend" of B's in the past, she now decided to block him in her buddy list. In real life this might lead to undesired social situations.
- Option 2: Buttercup's identity is defined in layers, where different peers might exposed to different aspects. Strangers and blocked parties might not be able to see her username, image, or other profile information, while friends, like W or D, are able to see those and realize it is their friend B that is also a part of the DinD group.
- Option 3: Buttercup could use a different unique identifier for her private friends and for her presence in the DinD group. In fact, she might use a unique identifier for each setting that she is ever part of, in essence enabling a different persona for each of them. She could of course link the same profile information to all of her personal entities, but then again, she has the option not to do so by default. If this was the case, V and I, might just see a different persona and not be able to link it to B's original digital persona.

## 4.2.3.3 Further Discussion and Resulting VPM Requirements

This scenario alone presented some very specific instances of groups and interaction, and can spawn a long discussion on these cases alone. In this part we touch just a bit longer on some additional points, but do not exhaustively explore and define this realm – that would be out of the scope of this initial presentation of the VPM theory and framework.

### *4.2.3.3.1 Enforcing symmetry*

One of the aspects to note is what it means to force symmetry of information exposure in our definition of the semi-public group. As we said in the beginning of the groups discussion, we define this type of group as one where anyone can see and add but has to expose their identity in order to see the identities of other members. A practical use of such grouping could be in the implementation of the "Gay Soldier Problem" scenario, described in 3.4.2. One way to implement this is as a private and closed group, but that might limit the option to meet new members who were not originally part of the clique. A semi-public (or semi-private) group definition might help deal with this issue. New members could join, but have to expose their identity in the process. A security person might jump out and ask what would prevent anyone, say the Sgt.'s army buddies, to create a fake avatar for the sole purpose of joining the group and exposing its members.

The answer, again, depends on the way the mechanisms are actually implemented. Here are three different options, do demonstrate:

- Option 1: Trusted third party has to vouch for the identity of new members. One way to implement this in a distributed scenario that does not assume a constant

Internet connection is to use one of the existing distributed certificate authentication methodologies, such as SPKI [D. Clarke et al., 2001].

- Option 2: In order to add one's self to the group, the user has to expose his or her physical presence. For example, having their device light up, or make a sound, or force the person to express some noticeable signal, such as ask for a password verbally.
- Option 3: The originators of the group accepted the possibility of such things happening when they set it as semi-private. Otherwise they could have chosen to set up it in a different way.

#### *4.2.3.3.2 Possible group types*

"TXTmob" is a project that deals with text messaging for protest swarms. Although it is infrastructure based (requires a server to manage different mobile phone text messaging groups), there are some parallels with our own distributed approach. TXTmob defines four types of groups that can be set by users who create their own groups. One axis was membership – whether it was open (public) or closed (private). Creators could also specify whether any member could send a message to all other members (umoderated) or whether only privileged users could send messages. This combination created four major types of configurations [Hirsch & Henry, 2005].

In our context as well, we can of various types of possible groups that might be useful for interaction, organized along various axes. Because we want to keep the VPM framework general enough for a large variety of uses, we have more than these four types of groups. As we can see from our discussion, in this context groups are very much synonymous with shared settings, so this discussion also reviews various types of shared setting mechanisms.

The originator of a group is defined as its owner or administrator, and he/she would be able to set the group's configuration. However, the owner may choose to give others some or all of the administrator's power. Some of the various options for group configurations are as follows:

- Explicit/Implicit enumeration of group members:

  *Explicit groups* enumerate their members. This might be better from the aspect of security, but harder to manage and deal with things like merges/revocations/synchronization between group members. There is a lot of work in this area though, and architectures such as UMI use it [Ford et al., 2006], so we can make use of these as building blocks for this type of groups. *Implicit groups* do not enumerate members - it is enough to present the right group credentials to "unlock" the group's setting.

- Axis of public vs. private – as we discussed in our example – some groups would be open to all, and some would be locked to members only, and even their existence would be hidden from view as much as possible. As we demonstrated with the semi-public group example, there are also options between the two ends.
- Moderated/unmoderated "write" permissions – similar to the TXTmob moderated/ unmoderated mode which defines who can send messages to a group's setting. In a small implementation note, this could be defined as a certain authentication

signature that the group's messages have to be signed by, in order for members to accept and handle these messages.

- Moderated/unmoderated "invite" permissions – defining whether anyone could invite others to join the group or only those with those administrator permissions.
- Serendipitous group is yet another type of group, where the group key is defined not by an explicit identifier, but by a more "fuzzy" definition. We introduce this idea in more details in section 4.4.3.

#### *4.2.3.3.3 Managing groups*

There has been a lot of work done in the area of distributed management of groups and permissions for example [D. Clarke et al., 2001; D. E. Clarke, 2001; Ford et al., 2006].The goal of the VPM framework is to provide enough flexibility for integrating with these existing types of groups and group management.

For simplicity, specific implementations that use the VPM framework may choose not to implement the full spectrum of flexibility, and restrain themselves to only a few types of groups and shared settings, in a way similar to the TXTmob approach [Hirsch & Henry, 2005].

## 4.2.4 Entities and Entity Networks

### 4.2.4.1 Blurring Boundaries and Rise of the Entity

Now that we went over the different types of concepts and justified their importance to the framework, we are going to blur the definitions of ideas like personas and groups and merge them into a single type of object that can take different roles in different places.
We will claim that shared settings, service identifiers, node identifiers and group identifiers can all be defined as similar *entities*.

As we discuss in section 2.3, nowadays our mobile phones have become almost synonymous to their users. From my peer's perspective, my mobile phone is the same as me. If they used some location based service that let them know about my whereabouts, they would most likely consider the location of my phone as my own location. Also, users do not usually care if their peers are connected and chatting through their phone, PDA or laptop. In this context, it makes more sense for the device to identify as the user himself, rather than the details of the specific device.

However when we look at a setting that includes only a single user's devices, communicating with each other, that distinction is very important. A user might configure his devices with a "behavior program" that exchanges social information between the devices to synchronize between them, or a program that makes sure key files are backed up on all of them. In that context, the user is the unifying setting, and the devices communicate among themselves within that setting.

In the same way, this idea could be expanded from users and their devices to users and the organizations they belong to. For example, "MITStudent" can be a group that its members are MIT's students, but it can also be an addressable identifier used by its members in when communicating with an external entity presiding in a higher hierarchy share setting. To

demonstrate this, the left side of Figure 4.11 shows an example where "MITStudents" acts as a group for the two users depicted. The group also serves as a *shared setting* for the two users, allowing their devices to discover each other each other and communicate through that shared setting. Within the setting, they each have their own unique identities, which are defined as entities as well.

The right side of Figure 4.11 shows an example where "MITStudents" would be used as a direct identifier of a host. In this example, one of the previous users, "Sam", wants to pass through a door. This specific door allows any MIT affiliate to pass through, with presentation of the appropriate credentials. In this scenario, it would be enough for Sam to identify himself as "MITStudent" rather than expose his personal identity.

This example could be extended in both directions. Sam might be able to identify himself as "MIT" towards an entity from a context that is outside the scope of the institute. Should it seem stranger when Sam can Identify himself as "MIT" to other entities than when Sam's phone identifies as "Sam"? The hierarchy and configuration of shared settings should be able to define whether sub entities can identify themselves as "the setting" both towards the outside and within the setting (using the setting's name within the setting could allow entities to perform *anonymous broadcast* within the scope of the setting. We touch on this in section 4.4.2).

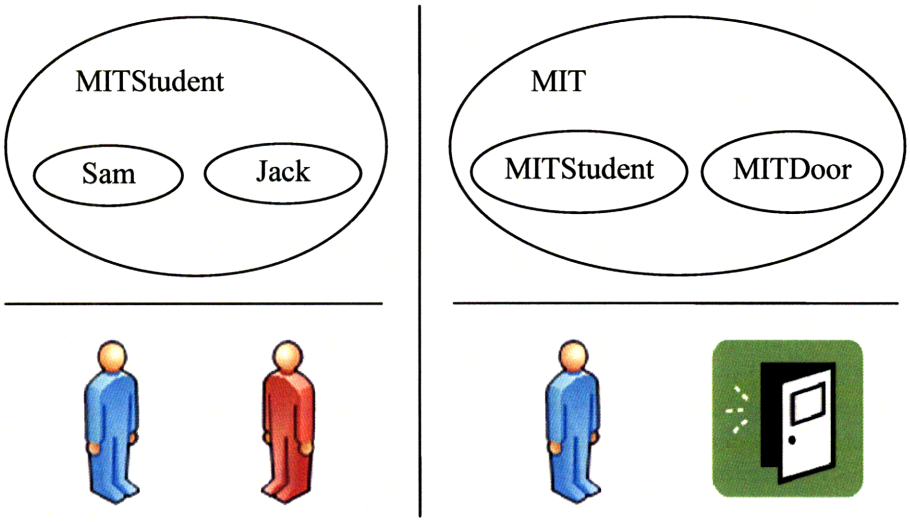



**Figure 4.11 - Examples for use of an entity ("MITStudent") both as a group identifier and as an addressable personal identifier.**

## 4.2.4.2 Naming Names

The following example brings our discussion to the need to define the naming schemes for the different entities and entity nesting and hierarchies. In this section we do not deal with the network level names, but with the need to have human readable mnemonic handles that represent the different entities and entity hierarchies.

In line with the Internet's domain name or WWW, it is important to use human readable names and aliases. We consider two types of naming styles:

- Option 1: Domain name style naming: As used also by Zeroconf [Williams, 2002], names are organized in a way similar to the following example: *"laptop.nadav._npp._tcp.viral.ml.mit"*. In this approach, named entities are arranged from the inside outward – the left side is the lowest level of the hierarchy, where the rightmost side is the highest level domain ("mit"). The example also shows Zeroconf's convention of marking predefined services with an underscore, aswell as the fact that the device providing the services comes after the service declaration, which makes for more efficient queries [Williams, 2002].
- Option 2: Directory style of naming, used by SPKI/SDSI [D. Clarke et al., 2001] identifiers. For example: mit/ml/viral/nadav/laptop/print_service.
- Option 3: A hybrid style similar to the way HTTP works today, that might allow for smoother merging with existing internet addresses or existing Zeroconf and SPKI implementations, for example: "some.domain.name/mit/ml/viral/nadav/laptop".

For the most part, in our discussion and examples we will use the second option, and the "/" delimiter to note nested entities.

Another aspect of the naming scheme is the issue of public and private names. As we discussed earlier, people make use of groupings and categorization in their mind. This internal categorization and naming may not propagate to the outside world. Also, sometimes users might want to use their own names for groups and entities, that is easier for them to remember, rather than use a public name that is set by an entity's administrator. Luckily, the SPKI/SDSI naming scheme supports local scopes of naming and their integration with external names using certificate chains. By using SPKI as a building block, the VPM architecture is able to accommodate this as well.

Another thing that we need to decide with regards to implementing the VPM framework is whether to enforce some conventions on the naming of different entities and entity hierarchies as Zeroconf does, or keep it unconstrained. The idea is that by marking different entities as different types, we could optimize for queries and searches, as well as management of the database of entities that each device must implement.

Our current tendency would be to first define the general version of the VPM architecture without constraining naming. Once this is done, we could describe a specific implementation of it that does support such naming conventions, which might be similar to this example:

**<group identifier(s)>/<personal identifier>/<device identifier>/<service identifier>**

We could define four "quadrants" to an address, where specific delimiters would mark different types of entities. For example:

mit.medialab.viral/nadav/cellphone/internet_access_service

## 4.2.4.3 Example Story: Part 4 –Entity Hierarchies and Identity

Let's return to to our gang. Actually for this part of the tale we focus on our loving couple, as Westley comes to the palace to pick up Buttercup on their way to a nice afternoon picnic. As we shift our attention towards some more familiar networking scenarios, we shall place several networked devices in the palace: Buttercup's laptop, a printer, and an Internet gateway point. As we assumed earlier, all of our devices will be part of the same ad-hoc

wireless network. The printer offers a network printing service, and the internet gateway a bridging service between the Internet and the ad-hoc network. Westley is carrying a mobile phone, which also has a WLAN radio allowing him to be part of our ad-hoc network as well.

At this stage let us also assume that we have two affiliation groups in our scenario:

- "Palace" – The Palace's private group, including all of the devices and entities that are affiliated with the palace.
- "BnWForever" – A private group that includes B and W.

In addition, all of B's personal devices are part of B's private group, which we'll simply call "Buttercup". Currently she has just one personal device in the vicinity - her laptop. In a similar manner, W also has a private group names "Westly", and the only device that belongs to that group in the vicinity is his phone. He does not turn on the DPR persona and the W persona is the only one that is active.

So, if we look at the different entities and hierarchies that are present in the palace right now, we get the following:

- Palace/internet_gateway
- Palace/printer
- Palace/Buttercup/Laptop
- BnWForever/Buttercup/Laptop
- BnWForever/Westly/Phone

As in the previous example, there is actually another hierarchy, the one that all entities belong to – the physical shared medium, which in our case is the ad-hoc network. It acts as the *"top setting"* for our scenario. If we really wanted to refer to it explicitly, we would have to add it to the entity hierarchy, for example: "TopSetting/Palace/internet_gateway". However, since it is at the root of all entity hierarchies and it is also external to the VPM domain and out of its control, it shall remain implicit in our current discussion.

The "Palace" group acts as a shared setting for the devices that belong to it: the printer, gateway, and B's laptop in our case. Similarly, "BnWForever" is a shared setting for B's laptop and W's phone.At the moment, the "Buttercup" and "Westley" entities include only one device each, the laptop and mobile phone, respectively. Note that Buttercup's laptop (or actually, the "Buttercup" entity and any entity associated with it) is part of two groups – the palace and her private group with Westley.Figure 4.12 shows a depiction of the logical groupings and entities in the above scenario.

Identities that some of the nodes can don:

- Under the Palace shared setting, the laptop can present itself as:
    - "Buttercup/Laptop"
    - "Buttercup", since it's also part of the "Buttercup" entity
    - "Laptop", where laptop is a unique identifier (or is associated with a unique identifier - UID).
- In fact, Buttercup's laptop can also be configured to present itself outside the boundary of the two shared settings, and expose itself directly in the TopSetting. It could use the above identities:
    - "Buttercup/Laptop"

- "Buttercup"
- "Laptop"

As well as:

- "BnWForever/Buttercup/Laptop"
- "BnWForever/Buttercup"
- "BnWForever/Laptop"
- "Palace/Buttercup/Laptop"
- "Palace/Buttercup"
- "Palace/Laptop"

Unless it was restricted by the owner of the group entities, the laptop node could even use its group entities as an identifier, as demonstrated also in Figure 4.11:

- "Palace"
- "BnWForever"

One may wonder if this is not an excessive redundancy, and in many cases it probably is. However, in an operation mode where security is very important, to support a goal of 'unlinkability'[Pang, Greenstein, McCoy et al., 2007], we might want to give each of these identities a unique ID, and possibly their own encryption keys.

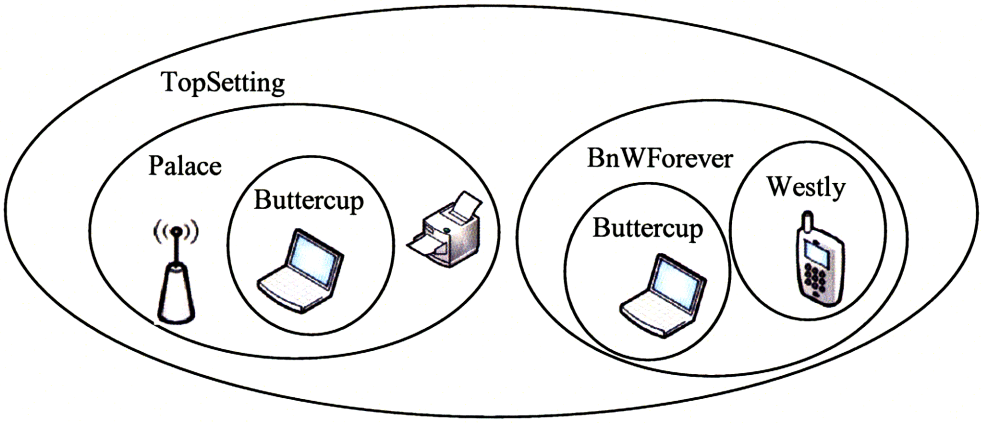



Figure 4.12 - Shared setting example – Palace simple network.

#### 4.2.4.4 Representing Complex Entity Networks and Hierarchies

As things get complicated, we need to have a better definition of the relationships between VPM entities:

Definition 1. An entity that is associated as a lower hierarchy of another entity could be considered as a *child entity* of the higher layer entity.

Definition 2. The higher level entity would be considered a *parent entity* of the first entity.

Since we do want to represent realistic scenarios, it is possible to have a *many to many* relationship between parent and child entities. There are also various ways in which one could illustrate and define the parent/child relationships between entities. Figure 4.13 shows

two ways for describing parent child relationships, and it also demonstrates how quickly things can get very complicated.

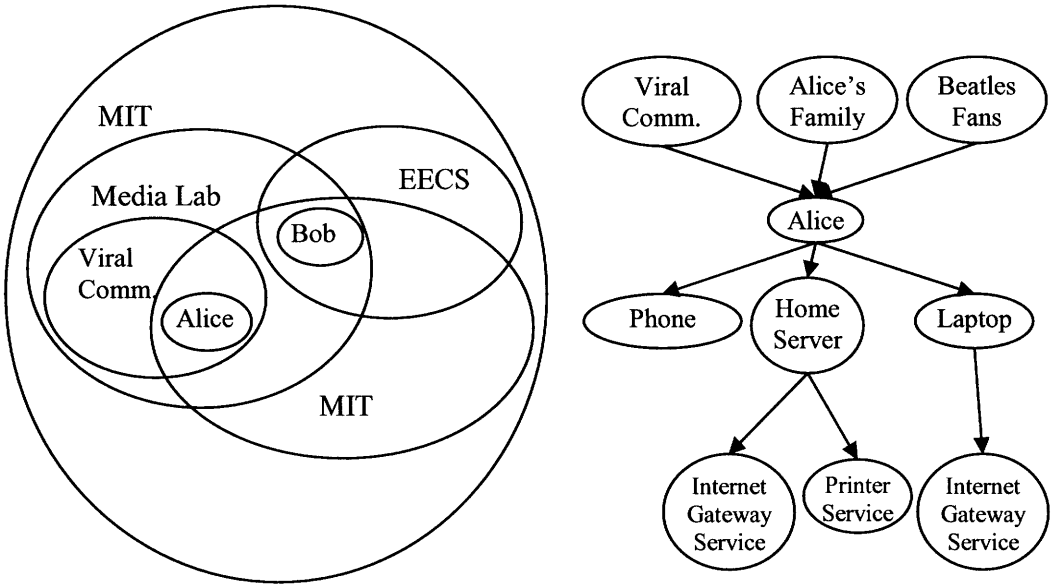



**Figure 4.13 - Two examples of representing entity relationships.**

In order to deal with this complexity, we shall convert the many-to-many representations to form where a parent entity can have many children entities, but each child can only have one parent. This means that if in the original graph a child entity has multiple parents, it will be duplicated in the resulting simplified representation. Figure 4.14 shows the simplified representation of the left side of Figure 4.13. This depiction somewhat reminiscent of the Collections interface [Viégas, 2000], however we need to remember that in our case this is just the omniscient view for the purpose of our discussion. Figure 4.15 shows the individual view of the entity hierarchies for each user. Remember that underneath the hood, each duplicate entity name would have its own unique identifier. One way to do it, might be that the identifier itself might be constructed from some hash over the specific hierarchy path. For example – the string "MIT/EECS/Bob" would get a different hash value than "MIT/MediaLab/Bob".

Aside from simplicity, this approach is probably something we would have done anyway, in order to preserve privacy of users as they participate in different shared settings associated with different groups, as we discussed in section 4.2.3.2.

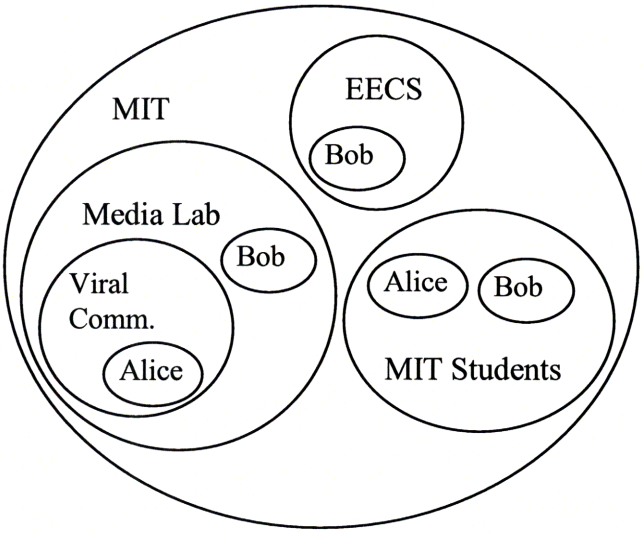



Figure 4.14 - Simplified version of the above example (left side of Figure 4.13)

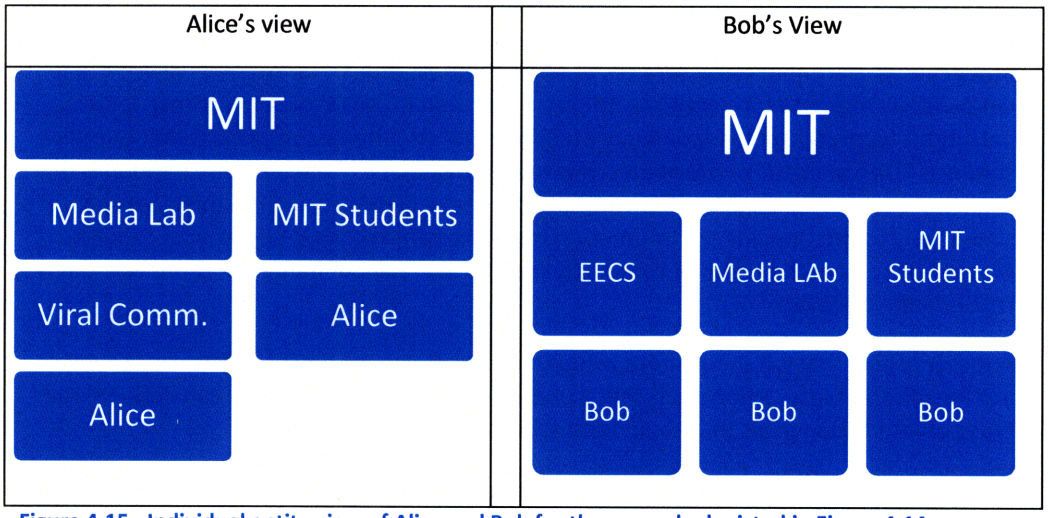



Figure 4.15 - Individual entity view of Alice and Bob for the example depicted in Figure 4.14.

### 4.2.5 The Discovery Game: Intersecting Personal Filters

One of the most important aspects of the VPM framework is the process of discovering peer entities and groups, and establishing the collection of active shared settings that make up the individual view of the network. We described it earlier as the overlaying of the set of personal filters of all entities that are part of the shared medium, or topmost shared setting.

In the "Private Authentication" paper, Abadi and Fournet talk about problems with current methods of private discovery and authentication: "In the course of authentication, a principal may reveal its identity to its interlocutor before knowing the interlocutor's identity with certainty. If A and B wish to communicate but each wants to protect its identity from third parties, who should reveal and prove theirs first?" [Abadi & Fournet, 2004]

We deal with the same issue in our framework, only on a multi-hierarchical level – each shared setting could serve as the common arena for discovering further peer entities and child settings that could be activated and drilled down further. This makes the discovery process a kind of a game (in the game-theoretic sense): In order to generate the set of available behavior settings that one might choose to join, one must expose information. On the other hand, exposing too much information may put the actor/persona at risk (i.e. the Gay Soldier Problem, section 3.4). This leads to the fact that on one hand, all entities would prefer to first listen and only then respond, in order to improve their control over the information and signal that they output. On the other hand, if all entities took this approach and just listened for the presence of peers, no one would hear anything and no peers would discover each other. This I also similar in notion to Goffman's idea of *identity negotiation* [Goffman, 1959]

We assume that services and other network entities appear and disappear dynamically. This could be caused by user mobility, moving in and out of range of peer devices. There are other causes that could cause services to appear and disappear, for example devices that deliberately decide to turn entities and services on and off, according to context, location, battery state, processor load, or many other reasons. Another assumption is that our VPM setting is "*Unmanaged*" or *distributedly-managed* by default – The basic assumption is that there is no central management or central database of identities and discovery settings, in a similar way to the Unmanaged Internet Architecture (UMA) concept [Ford et al., 2006]. Moving away from the realm of communication to the realm of humans – this is also the way that humans behave.

Another aspect that we should note, which directly affects the device's discovery behavior, are the different interaction goals that a user or a device might have. We can define three types of interaction goals that could be associated with each identity:

- Services offered – The device wants to offer a specific service to others.
- Services requested or needed.
- Opportunistic communication

Each of these goals would lead to different interaction – an entity offering a service would be more inclined to advertise its presence to potential clients, or respond to service requests. A device in need of a service or looking for a peer is more likely to willingly expose its presence, or at least its desire for the service. Entities looking for opportunistic communication might be more selective in how they advertise their presence and how they respond to peers' presence advertisements.

## 4.3 Definitions: Virtual Private Milieus

In the earlier sections we introduced many of the necessary concepts for understanding and justifying the need for the VPM framework, it is time to nail down more deterministic definitions to the key terminology that we are dealing with:

Definition 3. A ***VPM entity*** or a ***principal*** is an individual, process, or active entity whose messages are distinctively recognizable because they can be tied to a globally unique identifier. In our model, communication occurs between different principals. A principal might also be referred to as a *networked entity*, *communication entity*, or *addressable entity*.

Definition 4. The principal's identifier might be a cryptographic key, for example a symmetric key or a public/private key pair. In this case the principal's messages might be tied to the key having it signed or encrypted with it. Alternatively, the identifier might be a unique address, or the result of a one-way hash function, which would reside in the header of the message. In our discussion, we will refer to all of these types of unique identifiers simply as ***keys***, unless specified otherwise. Similarly to SPKI/SDSI [D. Clarke et al., 2001], it is convenient to say that the principal *is* its key.

Earlier in the chapter, with Definition 1 and Definition 2 we defined *parent* and *child* relationships for entities and principles.

Definition 5. An ***active entity*** is one that a human actor or a digital host device has decided to "turn on" and make available for interaction with another other entities. This means that as the host moves around, it checks for potential interaction partners with the set of its currently active entities. An entity being active does not mean the device will necessarily respond to every potential interaction opportunity – this will depend on the result of its *context filtering algorithms.*

Definition 6. ***VPM host***, or *host* for short, is a device supporting the VPM framework, data structures, and algorithms. It has the ability to support configuration and management of a set of entities. At the minimum, it must be configured with at least one unique entity identifier that is tied to the physical device.

Definition 7. Shared Setting: As we defined it in our discussion, ***a shared setting*** is a term encompassing both physical and logical interaction arenas. In the digital networking world we define it as logical segmentations of a shared medium, including the segmentation that is the shared medium itself. A principal can serve as a shared entity for its child entities when they are co-located as part of the same shared medium.

A shared setting is different than a fully fledged *behavior setting* as defined by Barker [Barker, 1978] and Wright [Wright & Barker, 1959], as they consider the full makeup the situation – the location, the actors, and the props that are in the area. In our context, a shared setting is more like the props and locale by themselves, more similar to Meyrowitz analogy of "place" and "walls" in which the interaction occurs [Meyrowitz, 1985].

Definition 8. ***Top Shared Setting*** or ***TopSetting*** is the highest and most inclusive level of a shared setting hierarchy, which includes the largest set of possible entities. This highest level will usually be defined along the physical limitations of the shared networking medium. Top Setting is also referred to as the ***root*** of the network.

Definition 9. ***Shared Setting Scaffolding*** encompasses all of the information relevant to defining an entity as a shared setting. It is implemented as metadata associated with a VPM entity that defines properties of the entity in its role as a shares setting. It includes configuration information like access and administrator permissions, encryption schemes and their associated parameters, or information that defines the lifespan of a setting.

A shared setting may be active or inactive, or dormant.

Definition 10. ***Active shared setting***: A shared setting is can be set as ***active*** when at least two child entities of the setting's entity must be part of an existing setting.

For example, if we wanted to establish an active shared setting $E_S$ which resides under the TopSetting, $E_{Top}$, we would need at least two other entities $E_1$ and $E_2$, to be:

- Active in $E_{Top}$
- Children of the dormant shared setting $E_S$
- Desiring to communicate with each other (mutual passing of each other's filters).

From these requirements, the minimal set of entity chain needed to activate $E_s$ in our example would be $\{E_{Top}\backslash E_s\backslash E_1, E_{Top}\backslash E_s\backslash E_2\}$.

The two entities would use the *VPM discovery algorithm* to learn of the presence of a peer entity that is also a child of $E_S$. They would then execute the shared *setting session initialization algorithm*, which will activate the setting. Once the shared setting is active under $E_{Top}$, other eligible entities may join or leave it, as well as spawn sub-settings as children of $E_S$.

We coin the term of *digital behavior setting* to differentiate from Blanchard's *virtual behavior setting* in the fact that the actors themselves are digital rather than humans. There is no special significance of our choice to use the word "digital" vs. "virtual" other than to differentiate from the existing. Another key differentiator is that these settings may not be necessarily "naturally occurring" as virtual behavior settings [Blanchard, 2004]. For example, when a user sets up a home network to synchronize between his personal devices.

Definition 11. A ***digital behavior setting*** is a naturally (e.g. emergent network) or artificially occurring (e.g. home or corporate network) social-like interaction space between networked agents. A digital behavior setting includes at least one active shared setting, and possibly more (parallel to the traditional "back stage" and "front stage" mentioned in the context of human interaction regions [Goffman, 1959; Meyrowitz, 1985]. Like a traditional behavior setting, it is composed of the shared interaction setting(s), the props, and the interacting agents.

Due to the nature of the digital realm, it is possible for multiple digital behavior settings to be active simultaneously, and for any agent to participate in multiple digital behavior settings. Like traditional settings, the digital behavior settings also have a behavior program that defines the behavior of its agents.

We might note that even the artificially occurring digital behavior settings are most likely spawned by traditional behavior settings and actually serve as props for the traditional settings – for example – a corporate defined network setting that facilitates and reinforces the behavior program of the company.

Definition 12. A ***hybrid behavior setting*** is a behavior setting that combines a physical interaction arena with a single or multiple digital arenas (shared settings).

The idea of ***hybrid behavior setting*** comes directly out of the *hybrid interaction space* which is a central piece of this thesis. We still do not fully understand this space and its full definition is a matter of further research, which we hope to initiate with this work. Even two

people in the same room chatting with each other through and IM program and talking at the same time are engulf themselves in a hybrid behavior setting.

Our use of digital and hybrid behavior settings will be more for discussion and analysis rather than a mechanism and algorithms for software implementation. For example, we might want to analyze digital behavior settings from the angle of self preservation, for example looking at a file sharing arena with these analysis tools developed by Barker and Wright. We might be able to say that setting attempts to preserve itself with a game-theoretic approach of tit-for-tat, and by enforcing ratios. If everybody "leeched", the setting would cease to exist. A successful social system for file sharing, in order to survive, would have to make sure that "leeching" remains at a low enough proportion.

We are almost ready to define a Virtual Private Milieu (finally!).
According to the Concise Oxford Dictionary:

> **milieu:**
>
> noun (plural **milieux** or **milieus**) a person's social environment.

Definition 13. A ***Virtual Private Milieu*** is defined as the social environment that emerges from the aggregation of all active digital behavior settings that a human actor or host device takes part in, generated at any given moment. It is private, since the emergent view is unique to the actor. It is virtual since it has no physical manifestation, as it resides in the digital implementation of the VPM framework as well as the mind of the human actor,

One can think of a VPM as an individually unique Virtual Private Network (VPN), or actually, a collection of VPNs, side by side and superimposed, representing each shared setting. We highlight 'individually unique', since each host will have his own set of VPNs according to his own social settings. Finally, these VPNs-like objects are dynamically created and managed distributedly, in an ad-hoc manner.

An important note: From the way we described our example narrative, it might seem as if an instant-messenger like communications application is a necessary part of the VPM framework. This is not the case, and the IM-like application was just used for illustration purposes. Such an IM program *may* be implemented on top of the framework, but the VPM framework is a standalone middleware between the network and user applications. Devices could use it to communicate with each other with no human intervention.

## 4.4 VPM Implementation

The previous section described the main ideas of the VPM theory and framework. VPMs arise from a combination of human need, theories of how humans operate, and an attempt to make it technically feasible even with today's technologies. In this section we shall lay out the generalized concepts of the VPM framework. As mentioned earlier, it is applicable to any shared medium, like 802.11 wireless radios, a shared Ethernet bus, or processes running on the same software implemented "virtual bus", like the D-Bus frameworks [see Freedesktop.org]. Most of our examples will be from the wireless domain since it is so prevalent these days, without limiting the generality of the concepts or solution. Hopefully this framework could become the basis for a new networking modality for the ad-hoc space

and eventually in the infrastructure space as well – configured autonomously according to user's social context and manageable by them.

### 4.4.1 VPM Design Principles

Based on the concepts described in section 4.2, we can define the main design principles for the VPM framework implementation. The VPM implementation framework aims to be implemented as a middleware layer, between the network interfaces and supporting user applications. As far as user interfaces and higher level applications that would be needed to propagate the VMP status and management up to the end users and administrator, they are out of scope for this chapter, but are part of the larger Comm.unity platform that is described in chapter 7.

Aside from the obvious mechanisms that need to be supported and were defined in the previous section (like shared settings, multiple VPM entities, etc.), following are the high level design principles that the VPM framework should support:

- *Self-configuration / "Zero-configuration"* – support a set of techniques that that automatically create a usable data network without configuration or special servers. They allow inexpert users to connect devices to the network and have them automatically configured for network operation.
- *Distributed system* – not dependence on centralized server, or infrastructure. The framework might support centralized and infrastructure based elements if they are present, but they should not be mandatory.
- *Medium independent* – The general VPM framework should not depend on any specific IDs, addresses, or mechanisms that exist in any one domain. Further extensions might create a specialized version for specific domains, where existing features would enable optimization and improved performance.
- *Minimize verbosity* - the goal of the framework is to minimize algorithm and control message "chattiness" by using techniques like caching, hashing, aggregating content, etc. This is important mostly for scalability but also for security aspects. We should note that there is a tradeoff between the sizes of message payloads vs. the number of individual bursts. In different situations one would be preferred to the other. Large messages might be more efficient in terms of overhead, but may be more prone to be lost due to packet collisions.
- *Security/Privacy Properties*

  The following security and privacy properties should be supported by the framework, although not all of them might be utilized by default. There is usually a tradeoff between bandwidth and computational power versus the level of protection, especially when encryption is involved. Eventually we might define different VPM "flavors" or modes of operation that would support different subsets of these properties. Hopefully, as applications are configured, a security assessment will be perform to asses realistic security threats and adjust these parameters accordingly.

  (Note: Some of the following are similar to the ones defined by the secure discovery and communication schemes described in section 4.5*)*

- *Message* integrity – Guarantee that message content was not tampered with [Pang, Greenstein, McCoy et al., 2007]
- *Confidentiality* –
  - *Content Confidentiality* –No node outside a shared setting should be able to determine the content of messages sent within the scope of the setting. The case of confidentiality for a unicast message between a sender and receiver is covered by using a private shared setting between the sender and receiver.
  - *Presence Confidentiality* – In a similar manner, we can define presence confidentiality such that no node outside a shared setting should be able to determine presence of nodes that are active in the setting.
- *Authenticity* - Sender and receiver should be able to verify each other's identity when this property is desired. In our context, there is both the authenticity of a sender's (or receiver's) identity, and also the authenticity of the shared setting's identifier.
- *Anonymity* – When this property is desired and enabled, the framework should allow, as much as possible, for the sender of a message to remain anonymous to both an outside listener and the receiver.
  - An idea relevant for some scenarios – *anonymity except to authority* - in this case anonymity is defined between any "regular" peers, but there are certain designated peers that have the ability to authenticate the senders. This scenario has been raised, for example, when thinking about vehicular ad-hoc networks where anonymity is desired between cars, but the police should have the ability to authenticate any vehicle [Boyen, 2007].
- *Unlinkability* – As defined in [Greenstein et al., 2008], we will refer to two levels of unlinkablilty between a sender's packets:
  - *Strong unlinkability* – tying 2 packets together (don't care)
  - *Long term unlinkability* -

- *User maintainable* – keep the system simple enough that regular users could utilize and maintain them. Users should not be unnecessarily exposed to low level technical information like IP addresses, encryption keys, or manual device "pairing" protocols (like in Bluetooth). Whatever can be generated automatically and in a way transparent to the users – should (for example private/public keys or session encryption keys). This property also includes feature likes:
  - *Human readable mnemonic handles* that represent the different entities and entity hierarchies, and are configurable by the users.
  - *Private/public naming aliases* - Support for external names as well as private aliases that will not leave the user's domain.

## 4.4.2 Communication Model

To understand the VPM communication model, we start by looking at a single shared setting in order to understand its basic communication and discovery mechanisms and terminology. We then briefly discuss the model for hierarchical configurations of shared settings.

### 4.4.2.1 Generic entity types in a shared setting

First of all, we need to remember that a VPM entity can take different roles, depending on the context. Still, in each of these roles, the role's ID remains the same as the entity's unique ID, dubbed also ***entityID*** or ***eID***.

For any shared setting, we define several generic types roles. We have the role setting itself, identified with a ***Setting ID***, or ***SID.*** The SID is the setting's addressable identifier listed from the root – the TopSetting. In a way, it is analogous to the *subnet* address in IP networking. Within the setting, each node has its own unique identifier, ***nodeID***. As we discussed earlier, the nodeID could be defined as setting-specific, and each node will keep track of multiple nodeIDs for each setting that it is part of. Each of these nodeIDs will be handled as an independent persona/entity within the setting and have its own state-machine, child entities and any other entity related configuration.

In the next section we discuss different ways to exchange messages within a shared setting. For the purpose of our discussion, we define the sender's noedeID as the source ID, or *srcID*, and the receiver's ID as destination ID, or d*stID* – as depicted in Figure 4.16. As we just mentioned, all of these, including the SID, are the same format of *eID*.

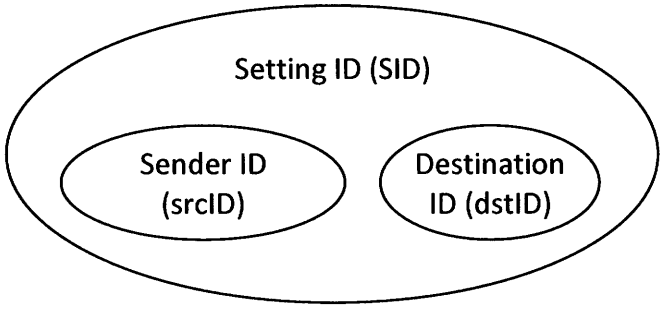



**Figure 4.16 – Generic shared setting entity roles and ID names. All of them are actually entityIDs.**

### 4.4.2.2 Message exchange

Within a shared setting, a message sent with source address set to srcID and destination set to the setting's SID will be received by all nodes that are part of the setting, thus implementing a message *broadcast* that is limited to the scope of the shared setting. A message sent to a specific destID within the setting is the analog of a unicast message. Actually, it is only considered a unicast at the level of the current setting. This is because that dstID may in effect be a group entity, shared by multiple nodes currently present in the network, or even active as a shared setting of its own, nested as a child within the current shared setting.

Figure 4.18 depicts these two basic types of message transmission. When observing the entire network from the scope of its TopSetting root, both of these message types are actually sent as multicast messages aimed at a subset of the nodes/entities that are present in the shared network medium. At this point we shall treat all messages in the network as unreliable transmissions, similar to UDP, that do not automatically elicit an acknowledgement message response from the receiver. This is due to the complexity of implementing a reliable transfer protocol when the sender does not have a clear idea of the actual number of receivers currently present. There are actually several ways to define and implement reliable message transfer for this scenario, but they are out of the scope of the current introductory document.

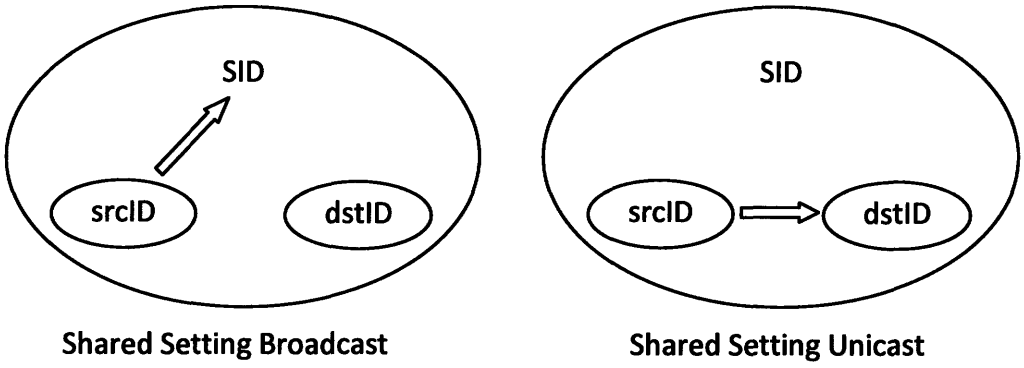



**Figure 4.18 - Broadcast and unicast addressing within a shared setting.**

Another kind of messages that could be transmitted within the shared setting is anonymous messages, where the sender identifies itself with the SID rather than its own srcID. In order for the anonymous transmission to work, the setting has to be configured to allow it, and also the different receivers have to agree to open up anonymous messages. Even if the shared setting allows it, users may configure the individual devices to ignore such messages to avoid undesired spamming. This is depicted in Figure 4.19.

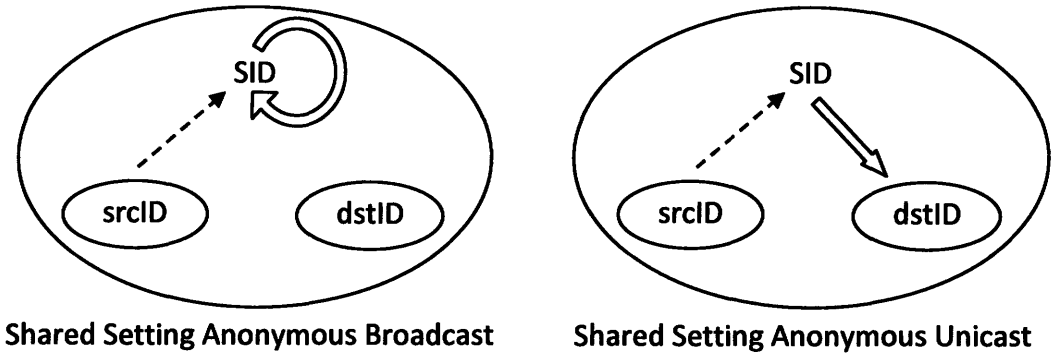



**Figure 4.19 – Anonymized Broadcast and unicast addressing within a shared setting.**

### 4.4.2.3Single Setting Message Primitives Example

Figure 4.20 depicts a very simple shared setting scenario, where the setting is defines by the "Buttercup" persona's entity, and it has two active child entities – one for B's phone and one for her laptop.


Figure 4.20 - Simple Single Shared Setting Showcase

Figure 4.21 shows the four message primitive mentioned above, where the entity addressing is absolute from the location of the top setting root (represented as '>'). Figure 4.22 shows the same configuration but with use of relative addressing (where '-' represents a null field). For brevity, we will use the relative notation for most of our discussion.

As advance further into the world of VPMs, we would shedding these traditional terms of "broadcast" and "unicast", since, as we will shortly show, they lose meaning in VPM's generic communication model.

| Message Fields | SID | Source | Destination | Payload |
|---|---|---|---|---|
| Broadcast | >Palace\Buttercup | >Palace\Buttercup\Laptop | >Palace\Buttercup | <MSG> |
| Unicast | >Palace\Buttercup | >Palace\Buttercup\Laptop | >Palace\Buttercup\Phone | <MSG> |
| Anonymous Broadcast | >Palace\Buttercup | cNum><record><rec | >Palace\Buttercup | <MSG> |
| Anonymous Unicast | >Palace\Buttercup | >Palace\Buttercup | >Palace\Buttercup\Phone | <MSG> |

Figure 4.21 - Message fields for the four types of shared setting messages with absolute addresses.

| Message Fields | SID | SourceID | DestinationID | Payload |
|---|---|---|---|---|
| Broadcast | >Palace\Buttercup | Laptop | -- | <MSG> |
| Unicast | >Palace\Buttercup | Laptop | Phone | <MSG> |
| Anonymous Broadcast | >Palace\Buttercup | -- | -- | <MSG> |
| Anonymous Unicast | >Palace\Buttercup | -- | Phone | <MSG> |

Figure 4.22 - Message fields for the four types of shared setting messages with relative addresses.

### 4.4.2.4 Additional Messaging Parameters

There are further types of messages or message parameters that could be defined under the VPM framework, which we will mention briefly in passing:

- Time related scope of the message:
    - Ephemeral – the message is sent to whoever is currently present and active in the shared setting.
    - Session-lasting – the message will remain "active" as long as the current shared setting session is active. New peers that join the setting will receive the message as well from its source or one of the other peers.
    - Persistent/"sticky" – The message will remain associated with the shared setting for longer than the session, until it times out or is proactively deleted.
- Reliability
    - Unreliable – no acknowledgement expected.
    - Reliable
        - Unicast – Acknowledgement expected.
        - Broadcast – using one of the existing reliable broadcast techniques.
        - *Pseudo-Unicast* – We can define a type of reliable message transfer where is enough to receive a predefined number of N acknowledgments, after which the message is considered as reliably sent. This is relevant for cases where a sender wants to make sure that at least one peer (or more) received a message, without caring that it be a specific peer or peers. For example, when Westly wants to send a message that would be successfully accepted by any of Buttercup's devices – it is enough to have just one device receive his message and acknowledge its reception.

### 4.4.2.5 Communication over Hierarchical Shared Settings

In this section we shall go over an example to get some idea of how the VPM communication would work in a hierarchical network of shared settings. Figure 4.23 is based on the same example depicted in Figure 4.14, however to make the example more interesting a second device is introduced to the "Buttercup" shared setting – Buttercup's mobile phone. Figure 4.24 shows the spanning tree topology of this hierarchical network of entities. As mentioned earlier, the entities that are children of multiple shared settings have unique keys, even though they may have the same human readable name and are part of the same host. To make sure this is clear, each of these entities is noted with its own unique subscript number. The single shared setting example from section 4.4.2.3 is nested within this example, and marked.

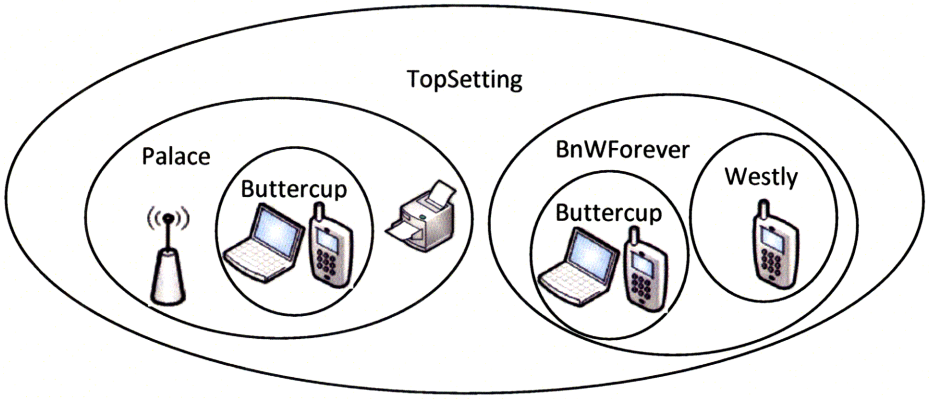



Figure 4.23 -Sample shared setting hierarchy - Palace example.

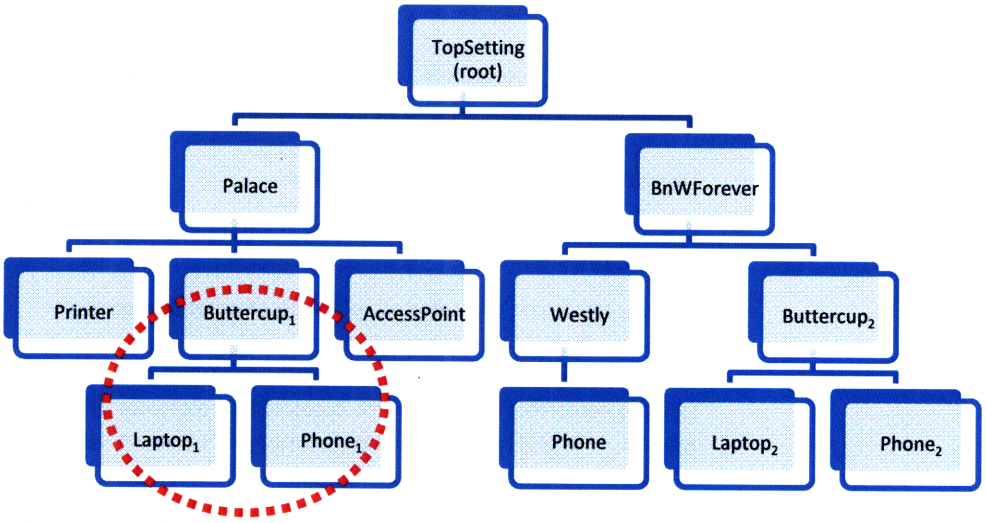



Figure 4.24 - Palace example, full entity network hierarchy. Circle marks the entities from single setting example from above.

Figure 4.25 gives an example of the scope of messages when dealing with nested entities and shared settings. Four messages are depicted, two of them we have already seen in the previous example.

- Source: $Laptop_1$ ; Destination: Phone

  ➔ Receiver: $Phone_1$ (and any other host that would have the $Phone_1$ entity or be defined as a child of it.

  In our example, since we have only one host for the entity $Phone_1$, it will be the only one receiving the message, and by traditional terms we would call it 'unicast'.

- Source: $Laptop_1$ ; Destination: $Buttercup_1$

  ➔ Receiver: $Buttercup_1$ (and any other host that would have the $Phone_1$ entity or be defined as a child of it.

In our example, since we have two entities that would receive this message – Buttercup$_1$ and Phone$_1$, since it is defined as a child of Buttercup$_1$ (notes as 'Buttercup$_1$/ Phone$_1$'). Incidentally, in our example, both entities are part of the same device, but this is arbitrary. If we look just at the transmission to the Buttercup$_1$ entity, it might seem like a unicast in traditional terms. If we look at the collection of devices that are children of Buttercup$_1$, this would seem like a broadcast, or multicast.

- Source: Laptop$_1$ ; Destination: Palace
  ➔ Receiver: Palace and all entities chat are part of the 'Palace' branch of the network tree.

- Source: Laptop$_1$ ; Destination: TopSetting
  ➔ Receiver: All entities in the shared medium.

Please note that these are receivers that are 'eligible' to receive the messages. Their hosts may *choose* to ignore these messages by filtering them out (We are aware that there is also the risk of denial of service attacks by "flooding" with unauthorized messages, and are incorporating a solution that takes this into account, however this is out of scope for this document)

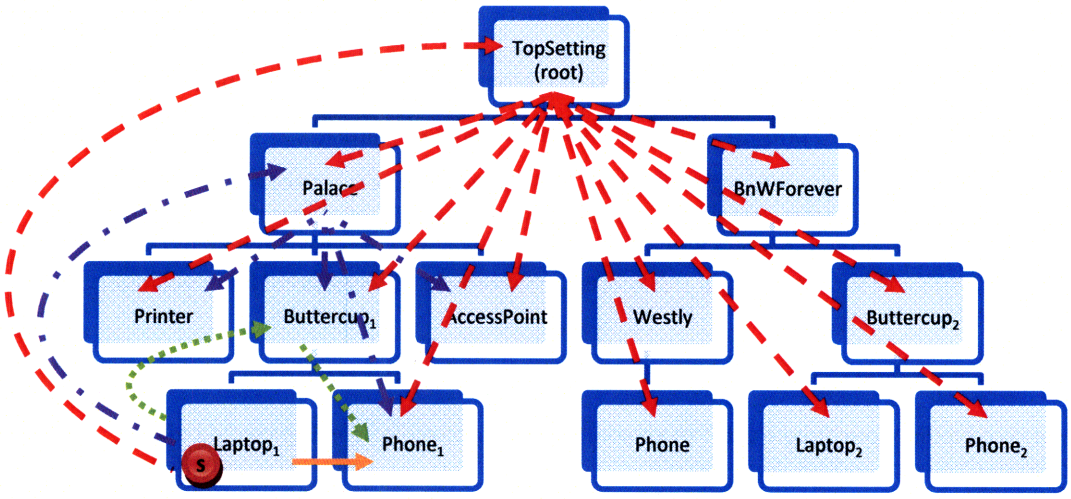



**Figure 4.25 – VPM messages in a multi-setting scenario. Arrows originating at Laptop$_1$ represent a transmission to the different destination entities. Respective output arrows lead to all child entities that would receive the message.**

As we show, even though all devices are part of one physical shared medium, we impose a subnet structure. In this structure, messages could be routed and processed according to subnet/host divisions, very much like the current packet processing algorithms for the Internet. We can even go as far as implement routing protocols, where the routing is performed within a device's VPM, forwarding messages between different shared settings. The algorithms that do it could be the same ones implemented in traditional networks, and we benefit from all of the existing work and theory in this area.

### 4.4.3 Identifiers, Network Addresses, and Address Matching

#### 4.4.3.1 Addressable Entities

In the theoretical concepts we defined a VPM entity, and described different roles it might take in different contexts. However, when implementing VPMs for practical applications, we can link some entities to specific real-world objects or network services.

Many times the principals that communicate with each other are linked to objects such as the following (but not limited to them):

- *A specific network interface*, for example a MAC address, IP address, or Bluetooth interface address.
- *A device* – a unified ID common to all of a device's network interfaces. E.g. Buttercup's mobile phone.
- *Personal identity* – name / pseudonym of a person (or persona) that is common to all of their devices. For example – "Buttercup/phone" and "Buttercup/laptop". Both also have the standalone identity of "Buttercup" and would be able to respond if anyone asks "is Buttercup around?
- *Group identity* – identities that represent a group, or a set of entities – "MIT students", or "people who attended the Media Lab's Fall 2006 open house".
- *Service type* – "printer", "display", "internet gateway", etc.

As we mention earlier, as we implement the VPM framework we can decide whether to structure the VPM addresses with some conventions that would mark out the above object types, to ease their identification and optimize their handling. One way could be to identify them according to the structure of the address, for example by using predefined delimiters and ordering:

**<group identifier(s)>/<personal identifier>/<device identifier>/<service identifier>**

We could define four "quadrants" to an address, where specific delimiters would mark different types of entities. For example:
"mit.medialab.viral/nadav/cellphone/internet_access_service". Using this convention, "mit.medialab.viral/ /cellphone/" would identify the two entities as a group and a device without forcing a personal identifier to be configured.

Another option is to distinct between different object types through the structure of the object address, for example by having a prefix field that indicates the entity type.

#### 4.4.3.2 Supporting Serendipity: 'Fuzzy' Addressing and Address Resolution

All of the above types of entity keys and addresses have been discrete, "noise free" identifiers which have to be generated by an originating entity or a third party, and distributed among all parties who need to be part of that identity or interact with it. For

example, if Buttercup set her network printer to work only for people from the group "Palace", then only nodes that have the correct "Palace" ID would be able to even see that it exists. These types of discrete identities are very good for interaction with people that we know in advance, or with relation to contexts and groupings that we know in advance. In most cases, especially when security is desired, in order to gain access to a group, its key has to be distributed to the members ahead of time, usually through a different (more secure) channel.

The VPM framework also supports encouraging more serendipitous interactions, based on a different kind of entities and entitiyIDs that are very much open-ended and do not have to be previously known by all sides of the interaction. A match between two of this type of ***fuzzy*** entityIDs is made through some matching function that purposefully allows for errors, hence the 'fuzzyness'.

Here are three types of such identifiers with example:

- *Free text identifier* – In this case the identifier might have an associated free text field, or itself *be* a text field. For example under a discretely defined group, say "news", one user may define an identifier called "sports news", another will define "Sports", and another will define "sport". String matching could be performed, with a predefined distance metric between the different identifiers.

  These identifiers could be treated as though they have passed through noise, and when determining if there is a match between an advertised identity and one of the user's own identities, the comparison would allow some error value or other distance metric that the users will be able to set. This way, users will be able to exchange content and communicate, even if they never met or exchanged information before. Another example is that under another discretely defined public group, called "music", different people could write strings representing bands they like or different types of music, and enable discovering strangers with similar interests as they pass by them.

Another domain where these "fuzzy" identifiers could come from might be some type of profile or 'fingerprint' that is generated through learning and analysis of various parameters, for example:

- *Reality mining based identifier* – Reality mining [Eagle & Pentland, 2006] or other methods for profiling users based on their behavior or any other type of measure could be made to generate an identifier that is a function of the user's profile. Then, these identifiers could be used to introduce between people with similar (or complementing) traits or interests.
- Music collection profile based identifier – Anita Lillie's project, "Visualizing Music" breaks up a song file to its musical components and compose a bitmap that represents a "fingerprint" of the song [Anita S. Lillie, 2007]. Another project, MusicBox, builds on these tools and principle components analysis to analyze a full music collection and visualizes all of the songs on a multi-dimensional plane [Anita Shen Lillie, 2008], as depicted in Figure 4.26. We could use these individual song representations or full collection representation to create a fuzzy VPM IN. Those IDs could be children of a discretely defined music related group. Then, as people who registered to that interest group walk by each other, the VPM mechanism

would find collections that are similar enough to one another. It would then be able to introduce the users, or simply allow them to listen to each other's music or swap legally distributable music.

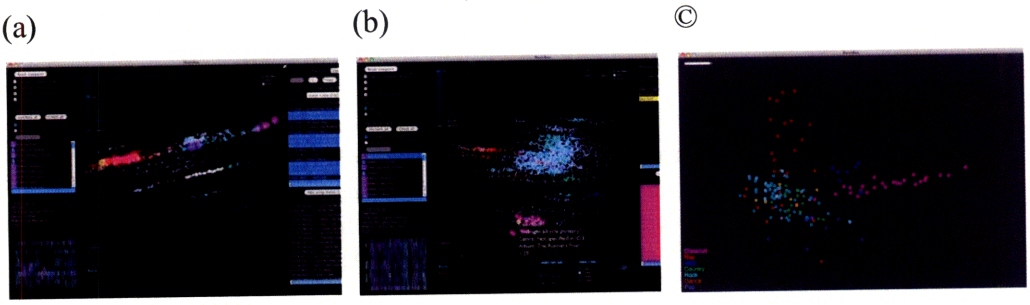



Figure 4.26 - Fingerprints of a music collection could be used as "fuzzy identifiers". Source: [Anita Shen Lillie, 2008]

### 4.4.4 Forthcoming Further Discussion

Due to the limits of this document, there are many concepts that have been left out and we would like to go back and define them at a later stage. The main thing that was left out is the detailed description and analysis of the VPM discovery algorithm which is defined in several flavors (from a naïve clear-text version to one that supports encryption and the building blocks described in 4.5. Additional concepts that were intentionally left out include:

- Shared setting operations (shared setting calculus?): Copy, merge, split, etc.
- Deeper discussion into the different types of messages and parameters.
- Shared setting life cycle: In [Wicker, 1987] there is a description of the life cycle of a behavior setting, depicting stages from "preconvergence", "convergence", "continued existence", to "divergence". This life-cycle analysis seems very relevant for integration with the VPM framework implementation. It also ties in with the need to define how new settings and sub-settings are spawned, and how one would decide how to pick the right parent entity to host the new setting.
- Discussion regarding persistent vs. ephemeral groups and settings.

## *4.5 Related Work: Networking and Security Building Blocks*

In this section we review some related work to our VPM discussion and include various tools and concepts that can used in building blocks for the VPM implementation, in areas of device and service discovery, decentralized naming mechanisms, security and privacy tools, and works related to combinations of discovery and security frameworks for wireless ad-hoc networks.

### 4.5.1 Device and Service Discovery

Service discovery protocols allow automatic detection of devices and services offered by these devices on a computer network. Their main goal is to simplify network administration

and promote user convenience, especially in dynamic and heterogeneous networking environments.

There is a collection of service discovery protocols that have been designed for infrastructure based networks, such as INS [Adjie-Winoto et al., 1999] or SSDS [Czerwinski et al., 1999] which usually utilize a dedicated *directory* service, located on one or more fixed hosts or a hierarchy of hosts. In a similar way to a Domain Name Service (DNS) directory, a service directory acts as a known rendezvous point, where service providers advertise their offering and clients send their queries and service requests.

Directory based systems are much less useful in ad-hoc, infrastructure-less, or very dynamic environments, which are characteristic in mobile computing and ubiquitous computing settings. In these dynamic settings, access to infrastructure is either not available or unreliable. Instead, there are two main approaches for distributed service discovery mechanisms, as also reviewed by [Campo et al., 2006], [Zhu et al., 2005] and [Pang, Greenstein, McCoy et al., 2007].

- ***Announcement-based***: In this approach, also referred to as a ***push-based*** or ***advertisement-based*** approach, service providers periodically send announcements of their presence and the service they provide. An example of this behavior is displayed by 802.11 wireless access points, which periodically advertise their presence and Service Set Identifier (SSID) for potential client nodes. Other examples implementing this approach are SLP's Service Agents (SA) that advertise services [Guttman et al., 1999], and the DEAPspace protocol [Nidd, 2001].
- ***Query-based***: In this approach, also referred to as a ***pull-based***, ***request-based***, or ***probe-based*** approach, clients search for services when they need them by transmitting request packets for those services. An example of a query-based approach is the Bluetooth Service Discovery Protocol (SDP) [Bluetooth SIG] which queries nearby devices for their offered services, or Apple's Rendezvous protocol, which broadcasts service queries using the mDNS protocol [Cheshire & Krochmal, 2006].

It is hard to find existing network discovery frameworks that encompass the VPM ideas in a comprehensive way. Zero configuration networking (Zeroconf) [Cheshire & Steinberg, 2005] and its likes [Campo et al., 2006] use distributed means to provide services that are similar to the infrastructure ones (for example IP address allocation or mDNS service discovery). In order to simplify the protocols, Zeroconf forgoes many security aspects, and broadcasts all of a devices identities and services to anyone that is part of the network, creating a dangerous security breach. This is in contrasts to human social behavior, where people choose to expose only a partial subset of their identity and affinities, depending on the situation and context. Other mechanisms, like [Czerwinski et al., 1999], do allow security but many times require a centralized server to keep the information and manage security measures like access control lists or encryption keys. Distributed schemes like [Ford et al., 2006] solve the distributed security problem, but are aimed mostly at connecting familiar peers who already know each other and had exchanged tight security measures through another medium (such as "introducing" devices when meeting in physical space).

#### 4.5.1.1 Bluetooth SDP

As mentioned earlier, the Bluetooth SDP [Bluetooth SIG] is based on polling for resources. What is interesting to note is that Bluetooth offers two types of queries: **Searching** for services, in which the device looks for the presence of a specific service, and **browsing** for services, in which the device looks for any offered service. These types of queries are relevant to goals of the VPM host or user.

#### 4.5.1.2 Zeroconf

Zeroconf [Cheshire & Steinberg, 2005] is a link-local technology, and works with link-local addresses and names. In the VPM case we do need such link-local addressing for starting up a VPM session, however it is key to have globally unique entity names that will persist as the host moves about. If we make use of a link layer's standard specs, we would make sense to use Zeroconf's mechanisms or Zeroconf-like mechanisms for generating a local link address at any given moment, which will be the address of the host. In some cases we could also set the host to have multiple addresses, for privacy purposes. However we would want an additional procedure to bind between the link address to the entities' unique addressing to enable communication.

If we use this scheme, then like Zeroconf, we could only guarantee security measures for the top-setting which are dependent on the security features of the top-setting's link technology. VPMs can only be responsible for providing security and privacy within its domain, which is from the interface layer and upwards. This might give us motivation to look for link technologies that are more secure than others, or run over a security mechanism that is link-layer specific, like SlyFi [Greenstein et al., 2008] which we discuss later in this section.

Like Zeroconf's declaration, VPMs must not be "any less secure than the current IETF-standard protocols" that it runs on [Williams, 2002, p. 4]p4.

Zeroconf's ease and simplicity come at great cost to security. In a video lecture given to employees of Google and available online, Zeroconf pioneer Stuart Chesire gives examples of Zerocon'f simplicity, and at the same time exposes its weaknesses in the context of our discussion [Cheshire, 2005]. Mr. Chesire gives an example and live demonstration of the ease of setting up a home security system with Zeroconf-enabled network cameras. He gave the example of simply plugging them in and giving them human readable names like "garage" and "kitchen". A viewer computer can discover and connect to them through Zeroconf, with ease [Cheshire, 2005, time: 29:16-31:50]. The problem is that from the talk it seems that ANY computer that gains access to the local network could access these cameras or at least know about their presence and their names. Any computer, that is, including a burglar's. A second security issue that came up in the video is the lecturer's demonstration of connecting is computer to the network at Google's facilities, and immediately found all of Google's Bonjour (Apple's version of Zeroconf) enabled printers in the building. Again, this is very useful from the aspect of simplicity and zero-configuration, but perhaps not everyone is allowed to print or even see the presence of all these printers, or any other Zeroconf enabled service. A secure discovery scheme like the VPM frameworks's proposal would prevent strangers and unauthorized devices from even seeing the existence of sensitive resources.

## 4.5.2 Decentralized Naming

### 4.5.2.1 SPKI/SDSI

Simple public key infrastructure / Simple Distributed Security (SPKI/SDSI) [D. Clarke et al., 2001; D. E. Clarke, 2001] is a is a public-key infrastructure that emphasizes naming, groups, ease-of-use, and flexible authorization. To access a protected resource, a client must present to the server a proof that the client is authorized. The proof is done in the form of a "certificate chain" that leads to a trusted principal, and proves that the client's public key is part of the groups that are on the resouce's access. SPKI/SDSI deals with two main types of entities – "keys" and "identifiers", the latter of which are primarily meant to be written in plain text and serve as a mnemonic handle for some human user. These human readable names are available to anyone who has access to the scheme's certificates. This property is also common to other schemes using human readable mnemonic handles, like Zeroconf [Cheshire & Steinberg, 2005].

The certificate chain framework, together with the ability to define private namespaces by users, could be very useful for our shared setting authenticated naming and addressing mechanism. The traditional SPKI/SDSI scheme does not deal, however, with scenarios where we do not want to expose such information, for example for security or privacy reasons. All of the human readable names in the certificate chain are exposed in plain text. As we discussed earlier, in some cases exposing membership in a certain group can be directly deleterious to a user, such an in the "Gay Soldier Problem" example depicted at section 3.4, as well as scenarios of free journalism under oppressive regimes, where a user's association with a certain group can lead to physical harm.

### 4.5.2.2 Unmanaged Internet Architecture

The Unmanaged Internet Architecture [Ford et al., 2006], or UIA, is an architecture aimed at providing zero-configuration connectivity among personal devices, using the idea of persistent personal names. UIA allows users to configure location independent personal names for their device without the need for a central allocation. It then attempts to create an overlay network between all active devices. This overlay network is encrypted and accessing required to connect to a peer that is already part of it using the right public/private key pair. The overlay provides routing abilities that connect between devices even if they are not physically proximate to each other (ad-hoc mode) and also through infrastructure, as long as a rout can be constructed through the overlay's devices. Personal names are created by having users "introduce" two devices locally, by a manual pairing process similar to Bluetooth pairing.

UIA is first of all useful to connect between a person's multiple devices. UIA provides a gossip and replication protocol to manage naming and group state required by the UIA user model, using concepts from the area of *optimistic replication* developed for database and file system implementations. Another interesting feature of UIA is that it uses the naming hierarchies as social network information, and leverages this for implementing a routing algorithm for the overlay.

There are multiple tangent points between the UIA and the VPM framework, but also many points of distinction. The UIA does make use of FtF interaction, but not as a goal but as a

means for secure device introduction and also to use peer devices as gateways to the rest of the overlay. It seems accurate to say that UIA deals with implementing a "*distributed infrastructure*" between trusted, familiar devices, rather than an infrastructure-less interaction framework. Any UIA device attempts to keep a constant number of TCP connections to other devices in the overlay, so long as it is possible.

A key difference between UIA grouping and naming is that all groups and all members are explicit. This is good for security purposes, but also adds great complexity with managing these groups in a distributed way. It seems that a large part of the UIA implementation and management data deal with managing distributed databases and dealing with distributing and merging changes and updates. Revocations of group privileges (for example when a device has been stolen or two people are no longer friends) are a very complicated process.

If trust is established between devices, it is absolute – one device will give the other all information that it has access to. In the example of introducing Bob and Alice's devices to each other [Ford et al., 2006, p. 4] page 4, UIA does not provide a solution to the case where Bob only wants to share his work phone and not his home phone with Alice. All devices in a group have the same consistent picture of the group. Hopefully we justified the human need for maintaining private and public aspects of social relationships, as well as asymmetric social links – features not supported by UMI.

Also, since UIA relies on Zeroconf [Cheshire & Steinberg, 2005; Williams, 2002] as its discovery mechanism, it is exposed to all of the privacy vulnerabilities of Zeroconf. *Had we tried to reenact the Gay Soldier Problem scenario using UIA, our poor Sergeant would be exposed.*

Finally, one of the key points of difference is that UIA is very good to discover and communicate securely with people and devices that the user already knows. In this respect, the VPM architecture could use UIA as a building block for those cases where strong security is needed which justifies the added complexity and overheads. However, UIA does not provide (and does not intended to provide) flexibility for interaction with strangers. The VPM framework is willing to relax some of the security features in order to enable much simpler group discovery and state management, as well as the option to interact with new peers and devices.

Another point where the two frameworks can integrate is by running the VPM framework over the UIA overlay network, which emulates a shared medium that can act as the VPM's top-setting!

### 4.5.2.3 Security and privacy aspects

There is a variety of existing security and privacy tools that could be used as building blocks for VPM security. Some of them might provide overlapping functionality, but each has different strengths and weaknesses. The idea is to eventually map out all of the possibilities and implications with regards to the security tools, and use the appropriate modules according to security requirements, threat analysis, and the tradeoff between the strength of the security vs. resources like bandwidth and processing power.

The VPM implementation makes use of very basic and tested mechanisms, described in detail in security literature, such as Bruce Schneier's *Applied Cryptography* [Schneier, 1996].

These mechanisms include: Private-public key infrastructure (PKI), which could be used as part as session initiation, symmetric key encryption that is computationally less intensive and is suitable for in-session encryption of data, or certificate signing which could be used for verifying message authenticity.

There are other, more advanced security and mathematical concepts that may aid as building blocks for the VPM framework:

- ***Group Signatures***, presented by David Chaum [Chaum & Heyst, 1991], provide a mechanism where only members of a group can sign messages. The receiver of the signature can verify that it is a valid signature from the group, but cannot determine which member of the group is the signer. In the case of a dispute, the signature can be "opened" to reveal the identity of the signer (by some calculation known to a more privileged type of member, such as a "group manager"). There are various derivatives of group signature concepts in the cryptography literature. An example of group signatures could be in an office scenario, where any member of a research group can print "anonymously" on a printer, but if he breaks it, then the manager has an ability to find out who it actually was. In a vehicular network scenario, it could mean that the identity of any car could be hidden from other cars, except the highway patrol, which has the ability to extract the unique user's identity.

- ***Ring Signatures*** [Rivest et al., 2001] are similar to group signatures in the sense that the receiver of a signed document can associated it with a set of possible signers but cannot tell which one was the actual signer. Unlike groups signatures, ring signatures do not need group managers and no setup process. Also, there is also no way to revoke the anonymity of an individual signature. An example scenario for ring signatures is a ring signature for a set including all senior Whitehouse officials. If one of them wanted to leak a secret, he could use this method without any ability to trace the identity of the sender other than authenticate that he is a senior Whitehouse official. A variation on ring signatures are ***mesh signatures*** [Boyen, 2007], another type of anonymous signature system, with a larger modularity that allows certificate chains to create more complex structure than rings and create more complex anonymous statements.

- ***Private Broadcast Encryption*** [Barth et al., 2006] is a mechanism for private broadcast with a strong privacy guarantee against active attacks, some thing that traditional broadcast encryption does not handle. A key feature of this mechanism is a constant decryption time regardless of the number of designated recipients.

- ***Private Authentication*** [Abadi & Fournet, 2004] presents a protocol that allows two mobile principals to communicate with each other when they come in contact without the danger of being tracked by third parties. One of the issues with private authentication, as the authors themselves point out (p.437), is that the current versions of the protocol do not scale well.

  In their discussion of Tryst's feasibility in the context of encryption protocols overhead, Pang et al. [Pang, Greenstein, McCoy et al., 2007] note that it is unlikely that clients will use more than a few services of a single type. As example they give example the OSDI 2006 log which showed that 90% of users probed for at most 12 unique 802.11 SSIDs [Chandra et al., 2006]. Pang et al. also state that "due to the

structure of social networks, the average user is unlikely to establish many direct relationships" [Pang, Greenstein, McCoy et al., 2007]. Although we do not necessarily agree with the last statement, we could use this insight to formulate a statement that is more technical in nature and less sociologically committing: We assume, in the context of VPMs, that there is a limited number of shared settings that are direct children of the root top-setting. This is important since in the VPM scheme any host may have a large collection of active entities as well as peers and services that it might want to interact with. However, even in case where the number active top-setting children is larger than we would like, there are additional methods to aggregate and group entities in a way that would reduce the overhead associated with using private authentication protocols, for example by using a ***bloom filter*** [Bloom, 1970]. It is enough that a peer ,device belongs to at least on top-level setting as our host device in order to rendezvous with each other, and continue the "drill down" discovery process. This assumption enables us to make use of overhead intensive but sometimes necessary security protocols.

- ***Private Set Intersection*** [M. J. Freedman et al., 2004; Hazay & Lindell, 2008; Kissner & Song, 2005] presents a very interesting solution to the problem where two sides have a private set that they do not wish to fully share with the other. They only want to find out which objects are present in both sets. Private set intersection algorithms give a cryptographic way to do this. One of the main goals of these mechanisms is for finding intersection between very large datasets, where the main challenge is in scalability. In our context, however, we might be able to use these methods as part of our discovery process. For example, by having two hosts who want to find their set of intersecting entities and groups without exposing any non-intersecting group. This could be used in a solution for the Gay Soldier Problem. In addition [M. Freedman & Nicolosi, 2007] builds on these methods for an algorithm to verify social proximity between parties while exposing minimal information about the parties' social contacts.

- ***The Onion Router (TOR)*** [Cerf et al., 2007] is a project that provides anonymity while using the Internet and helps defend against traffic analysis. Tor protects data by "bouncing" packets around a distributed network of relays. "To create a private network pathway with Tor, the user's software or client incrementally builds a circuit of encrypted connections through relays on the network. The circuit is extended one hop at a time, and each relay along the way knows only which relay gave it data and which relay it is giving data to. No individual relay ever knows the complete path that a data packet has taken. The client negotiates a separate set of encryption keys for each hop along the circuit to ensure that each hop can't trace these connections as they pass through."[Cerf et al., 2007]. In our context, we might think of scenarios where in order to gain added anonymity, we could "bounce" messages between entities and shared settings, so that it would be hard to trace the entity and setting that the message originated on. We can go even further and decide to even bounce the message between settings within the same device, in order to provide additional 'onion layers'.

## 4.5.2.4Secure Discovery for Ad-Hoc Interaction

### *4.5.2.4.1 Tryst and 802.11 Discovery Vulnerabilities*

Tryst is an architecture for enhancing confidentiality for service discovery mechanisms. It deals mostly with service discovery for 802.11 networks that provides confidentiality and authenticity. In [Pang, Greenstein, McCoy et al., 2007] Pang et al. introduce Tryst as well as an analysis of sensitive information that is exposed through existing ad-hoc service discovery mechanism. [Pang, Greenstein, Gummad et al., 2007] gives examples of careless exposure of information in existing implementations of 802.11 access-point associations in Windows XP and OS X wireless clients. In these implementations, in the process of attempting to discover any wireless network the device publishes a list of preferred SSIDs, which many times have a clear-text meaning, and can be used to identify the owner, as demonstrated in that paper.

We have independently developed a secure discovery mechanism, with several features overlapping with Tryst and Smokescreen [Cox et al., 2007] (discussed later in this section), specifically the symmetric key protocol. However, since both projects are further ahead in implementation, and in addition already provide some of the security analysis for these protocols, we will try to make use of them as building blocks as much as possible, and only make changes when add new features required by the VPM framework or for enhancements who's benefit is distinctly visible.

Tryst seems to be very much about unique 1-to-1 secure connections, about "me" and services/devices I know and. The VPM scheme does not deal solely with private communication between two trusted entities, but in creating trusted virtual shared spaces where different entities could feel free to expose themselves to some degree, for example – for advertising services to peers who are not necessarily known in advance, but are part of a larger yet limited trusted context. Another example is polling for any device providing a specific type of service, without the need to have it known in advance. In essence, we are somewhat relaxing the security requirements within the 'walls' of a trusted realm, in order to gain additional needed services. However, outside of the trusted realm, adherence to the strict security measures remains.

Tryst's starting point is at the analysis of threats in current network implementations and service discovery mechanisms. It then deals with answering those threats but under the existing framework, for example, Tryst suggests to wrap itself around the existing discovery methods, which makes a lot of sense for many reasons.

The VPM framework, in contrast comes from a top-down methodology: It is inspired by a human cognitive approach, attempting to understand how humans operate socially, and trying to mimic or reflect it in design of digital means of communication. It deals less with patching open vulnerabilities and more with how things might be done better. Also, most of the framework and concepts are not limited to specific standards or even link layers of general wireless networking, but to any entity in a shared medium.

#### *4.5.2.4.2 SlyFi*

Greensten et al. propose and implement a prototype of SliFi, a wireless link layer protocol for 802.11 like networks whose goal is to increase privacy. The key problem that is dealt with by SlyFi is the complete removal of explicit identifiers from wireless transmissions for parties other than the intended recipients [Greenstein et al., 2008]. SlyFi is made up of two layers – an implementation of the Tryst architecture for discovery and binding, and Shroud for protecting data transport. The overhead costs of SlyFi over WiFi are about 10%, which is

acceptable for the price of security. We expect that the overhead of a VPM implementation would not be more than this, especially if we build on this work and adapt it to our context.

There are some points to note with respect to the VPM framework discussion. First, as Tryst, SlyFi deals with a single identity per device. It seems that there is somewhat of a conception that address = device, whereas VPMs deal with multiple entities and identities per device.

Most of the examples given are for WiFi access point and infrastructure mode scenarios. Most of the security related problems and examples, as well as performance comparison, are with WPA, an infrastructure mode security mode. From the paper it seem shat the SlyFi link later is designed to replace that of 802.1 for *managed* wireless connectivity between clients and APs. It might be possible to adapt it for ad-hoc mode in a simple way, but we feel that by integrating SlyFi's key principles with VPM architecture, there would be a much greater gain in usability than a standard ad-hoc mode.

Like Slyfi [Greenstein et al., 2008], by implementing an encapsulation of VPM packets within 802.11 management frames that are ignored by normal 802.11 devices, VPM enabled devices can coexist with existing 802.11 devices. This also means we can have coexistence of encrypted and open VPMs (that use the existing link layer protocols).

#### *4.5.2.4.3 SmokeScreen*

SmokeScreen [Cox et al., 2007] is another proposal for a secure and private peer discovery and presence sharing protocol. It suggests some features that could be useful for our discussion, and it has some implementation ideas that are common with SlyFi [Greenstein et al., 2008], for example the use of symmetric keys for session encryption and temporary addresses, though some of the base assumptions for these two projects is somewhat different. Aside from focusing on secure sharing of presence and does not provide private data communication or deal with link layer MAC addresses. There are some other issues that could be noted, that help motivate our discussion of VPM's features: First, there is some assumption that within a clique, every member trusts every other member in the same way. VPMs allow for granular, individual, and asymmetric trust even between members of the same interaction group. In addition, the paper does not mention how to deal with click management and dynamics– like handling and propagating revocations, for example.

The part of SmokeScreen that is fully peer-to-peer is focused on individuals and clicks that know each other beforehand. In order to meet strangers, there has to be an online server involved, which means a necessary data connection to the Internet or another infrastructure network. The online connection may provide increased security, but it limits the usage cases and also the level of serendipitous interactions, which we care about so much. In VPMs our solution for this privacy issues is achieved by the use of a multitude of identifiers for a user, where even if the presence of one might be exposed, it still would not expose the real persona of the user, for example. Even though no evaluation of scalability and bandwidth usage efficiency is provided, its implementation for PDA devices it is still very positive. This demonstrates an existing implementation that uses security and encryption mechanisms and can run on mobile devices.

# 5 The Social Dashboard: Human-Centric Network Interface

*"To be trusted is a greater compliment than to be loved."*

*-George MacDonald, 1877*

The Social Dashboard, or SD, presents a novel concept of human-network interface, as well as device management interface. It is our proposed solution for an interface aimed at Face-To-Face Networking (FtFN): it allows users to discover peers and services in their social and physical vicinity based on the VPM architecture, and communicate with them using a provided set of interaction tools aimed to support face-to-face digital interactions. In addition, the dashboard is also an interface that allows users to configure and control the behavior of their device towards these peer devices and services, based on their social connection to the owners of the devices, and based on social metaphors in general. Similarly to the VPM framework, the general design approach is based on our findings and recommendations regarding the Hybrid Interaction space (see section 3.5). The SD is based on ideas from the area human centric design and cognitive load theory. Rather than attempt to lay out peer entities as they are laid out in the physical world, we try to represent them in a radar-like interface according to how they might be laid out in the user's mind – along axes of social relationships and trust.

## *5.1 Related Work*

### 5.1.1 Designing for Humans

The effect of cognitive load introduced by user interfaces is discussed in depth in Oviatt [Oviatt, 2006]. Oviatt makes a strong case for considering a user's unconscious cognitive abilities as a primary design principle for building user interfaces. A theme recurring throughout Oviatt's discussion is that "human centered design which minimizes users' cognitive load can free up mental resources, permitting us to perform well while also remaining attuned to the world". Oviatt lists eight user-centered design principles associated with improved human performance. Although Oviatt's work referred mostly to the design of multi-modal user interfaces, especially for educational purposes, they seem very relevant to our own interest in social interfaces for the hybrid space. The principles include developing interfaces that:

- Leverage from users' experience, knowledge, and engrained behavioral patterns, as well as adapting to users' behavior and preferences.
- Support users' natural and flexible multimodal communication patterns.
- Transparently guide users' input to minimize difficult sources of linguistic and behavioral variability, so system errors are reduced and usability is enhanced.
- Minimize cognitive load associated with user input planning (e.g., producing lengthy sentences).

- Accommodate users' existing familiar work practice rather than attempting to change it.
- Support representational systems as part of the interface (e.g., linguistic, diagrammatic, symbolic, numeric) that users need to perform their task.
- Minimize cognitive load associated with extraneous complexity of system output (e.g., unnecessary features that distract users' attention when completing a task)
- Minimize interruptions (i.e., whether due to distracting system features or explicit system interruptions), which undermine users' ability to engage in high-level planning, integrative thinking, and problem solving

Oulasvirta et al. studied the cognitive load when walking and interacting with a mobile device [Oulasvirta et al., 2005]. They observe the great fragmentation of people's attention resources when moving about the world and interacting with a mobile device at the same time, compared to more static situations such as laboratory or office settings. Their results show that "resource competition is very real and seriously constrains mobile interaction. The data conveys the impulsive, fragmented, and drastically short term nature of attention in mobility." (p.926)

These findings are relevant to two aspects of our discussion – First, improving the connection between the user and the network, like its status, topology, available services and devices. The second aspect is improving the connection between users to one another. A general comment that we can make regarding the Hybrid Space is that since the physical realm is directly related to the digital realm, a good design approach would be to unify as much as possible the attention resources needed to orient and deal with the physical environment and the mobile applications, which are linked to the physical environment.

### 5.1.2 Visualizing Peers in Local Proximity

We very briefly review related interfaces that display peers and network devices in a certain proximity. We shall only relate to interfaces that deal with relative distance that is generated by network data, and not interfaces that have access to absolute GPS coordinates.

Several applications use an interface that tries to lay out peers with some notion of their physical layout in the real world. For example, arrange them in concentric circles around the user, through some radar-like interface. The distance of peers from the user attempts to mimic their true distance (for example, by attempting to correlate signal-strength measurements with physical distance, assuming the signal strength deteriorates with distance). This approach is known to be very inaccurate since signal strength does not directly correlate with distance, as it may bounce off walls and degrade due to noise and physical topography. For more information about this, see literature regarding indoor positioning systems via radio signals. Another approach is to lay out peers around the user according to logical distance in the network, meaning the number of hops that a peer is away from the user. A peer that is one hop away will be displayed closer than a peer that is two hops away or more. An example of an application that uses this is the Cerebro presence interface [Ypodimatopoulos et al., 2008]. Jabberwocky [Paulos & Goodman, 2004b] has an interesting interface that displays "familiar strangers" whom the user encounters along a timeline. The user cannot interact with the peers or even correlate a peer representation with the physical person (for anonymity purposes). However, the user gets a sense of how many times they interact with the same strangers, and how long ago.

*The idea behind the social dashboard is different. Rather than attempt to lay out peer entities as they are laid out in the physical world, we try to represent them in a radar-like interface according to how they might be laid out in the user's mind – along axes of social relationships and trust. To the best of our knowledge, no other similar interface has taken this approach before.*

It should be noted that most of these interfaces are used for visualizing information for the user, and since they are aimed to represent a physical/logical real image that exists outside of the device, they are not malleable by the user. The Social Dashboard, however, in its role as an extension of the user's cognitive picture, is very a much malleable and bidirectional interface that not only presents data to the user but also receives valuable information from the user.

## 5.2 Design Concepts: Social Dashboard

In this section we review some of the ideas behind the Social Dashboard. Figure 5.1 provides an illustration the SD interface concept. It depicts a fictional scenario that shows different concepts that are part of the SD's design.

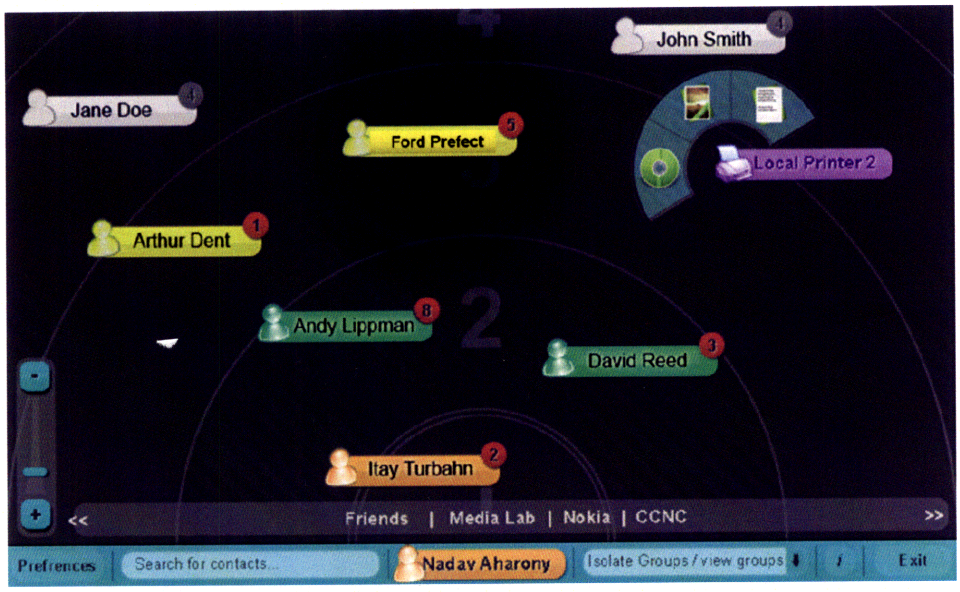



**Figure 5.1 - Social Dashboard concept illustration, made with assistance of Itai Turbahn.**

***Visualizing Social Distance*** - The main feature of the SD is that it visualizes peers, groups, devices and services on a scale of social distance. The main dashboard representation in the current iteration is a 'radar-like' interface, where the distance from the hub represents social distance, or trust distance, rather than physical proximity from the user, whether it be absolute or approximated according to "radio distance". The user is represented at the center-bottom of the screen ("Nadav Aharony" in the Figure 5.1 example). Outwards of the user emanate "Circles of Trust". The circles represent discrete boundaries for trust levels. These discrete separations are used for mapping each circle to different behavior. The specific behavior would depend on configuration by the user or programmer of an

application that uses the SD. Just as an example, it might be configured that the user will automatically expose their presence to entities in the closest circle and automatically accept content from them. Entities residing in a farther circle might see the user as "busy" or "away" even though the user is active, and any content sent to the user might have to be manually confirmed. Strangers might be in the farthest circle of trust (i.e. "no trust"), which could mean that the device will automatically reject any content sent, and might not even expose its presence to them.

- ***From the User Outwards*** - As we discuss earlier, and in line with the human centered design approach [Oviatt, 2006], we construct and present the user's social network from the single person's individual perspective. Our approach is inspired by the way we use our engrained senses, and put the user in the center of the interface. Figure 5.2 shows an early sketching of the SD interface, where the user is represented graphically at the center of the interface. Shortly after we moved to a layout where the user representation is placed the bottom-center of the screen, as depicted in Figure 5.1 with the "Nadav Aharony" icon. This was done for several reasons: The first is related to the way the device is held by the user. We wanted to create the notion that the "me" of the interface is closer to the user's body as they hold the mobile device. The idea is that the peers are places from the person outwards (even though their distance from him is not physical). The second reason was that this allowed us increase the width of the rings that lie between the circles of trust, allowing larger icons or more peers to be represented.

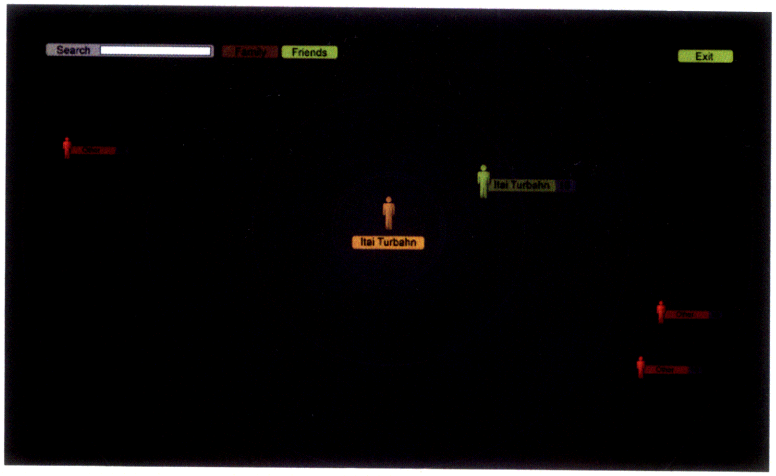



Figure 5.2 - Social Dashboard initial sketch. Actual image made by Itai Turbahn.

- ***Group and Context Based Filtering*** – As discussed in detail with respect to the VPM framework, a lot of our interaction is done with groups and sets of peers. In the same way that a person can "tune" himself to speak to a group of people rather than a single person at a time, the dashboard interface allows to view groups and interact with them directly. Messages could be sent to a whole group by directing it to a group icon, and a file may be transmitted to all members of a group by dragging it onto the group. Group icons as represented in one of the version of our implemented prototype are highlighted in Figure 5.3. A peer can be invited to a group simply by dragging him unto the group's icon. Aside for enabling

communication primitives and actions that can be performed for group entities, the SD concept also enables group based filtering which can be automated or manually operated. For example, the user may manually choose to see only known peers, or peers that belong to a certain high level group entity. This is depicted in Figure 5.4 where only entities related to the "Media Lab" group are highlighted and active. Automatic filtering might be useful especially in cases that there are many entities in the user's vicinity. In order to reduce the cognitive load, the system may highlight those peers and groups that are socially closer to the user, or according to the device's current active context(s). For example, the dashboard may be configured to show no more than 20 peer entities. In this case the system will show individuals that are in the users' closer trust circles, and might aggregate the rest to their respective groups or even generate new groups according to different clustering algorithms (e.g. group strangers together according to the number of times the user has been next to them).

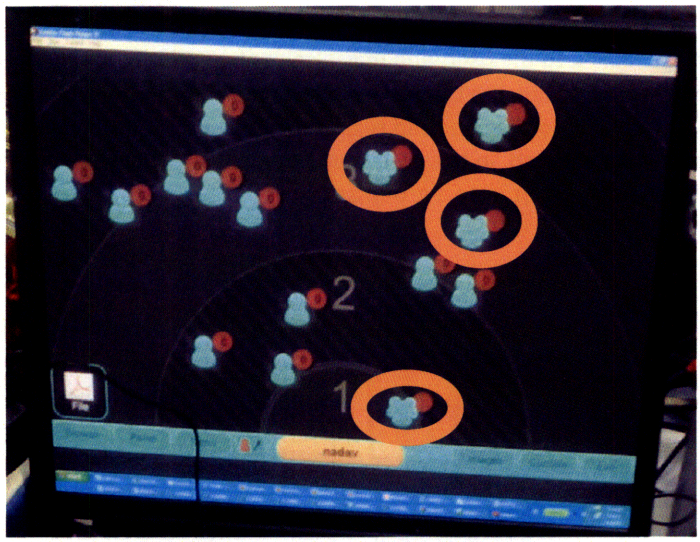



Figure 5.3 - Group entity icons in implemented prototype.

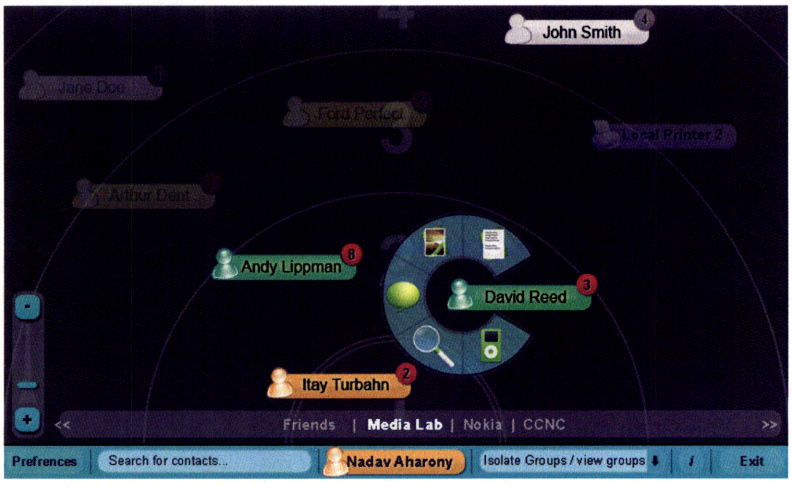



Figure 5.4 - Group based filtering. Image made with assistance of Itai Turbahn.

- **Context Based Actions and Services** – As depicted in figure Figure 5.4, we could implement a context-based menu that is turned on as the user clicks on the peer icon (and we have). The actions that are available to the user are defined by the nature of the peer entity and by the services that it wants to expose to the user's entity (according to the VPM settings). A printer for example, would expose a "print" icon in its action menu, while a human peer may have a "chat" icon in the action menu.

- ***Enabling Intuitive Digital FtF Interactions*** – One of the useful features when running the dashboard on a mobile device equipped with a touch-screen is the easy and intuitive ability for digital interaction with peers. For example, as a user walks around in a job fair or conference, she may meet a person. That person would immediately appear on their dashboard (probably as a stranger, in the lowest trust level). In an analogous way to giving the peer a physical copy of her CV, she may simply drag the electronic version of her CV onto the peer's icon, and initiate a file transfer. Note that the peer may only choose to expose their presence after physically talking to the user, and may also configure his own device to ask for his approval for accepting the CV file. This is illustrated in Figure 5.5. In a similar way, the user may talk to the peer and realize they have a common friend, or they might decide to stay in touch. In this case they can manually "drag" each other's icon to a closer circle of trust or unto a group entity.


Figure 5.5 - "Giving" data in the FtF space

- ***Enabling User Configuration*** - As mentioned earlier, the user has the ability to manually control the placement of his peer entities, and by doing this he can configure and control the behavior of the device.

- ***Interacting with the System's AI*** – One of our goals is that the system would eventually employ artificial intelligence (AI) or machine learning algorithms to try to learn the user preferences and social relationships in order to improve its performance. Automated configuration would also reduce the load of manually

configuring the information. A very simple algorithm, for example, might try to infer social trust or relationship according to the number of times that a user is around a peer, combined with the actual interactions that the user has with that peer (like chat messages, files transferred between the two, being part of similar groups, etc.). If the user allowed the system to change the behavior toward that peer automatically, the user would be able to see, for example, the peer "drift" closer from the outer rim towards the hub. If the system has made a mistake, the user could very easily see it, and also correct the decision immediately by dragging the peer entity back to where they think it should reside. The user's response could be integrated as a reward or punishment signal by the AI algorithm. There are many other ways to implement the AI's decision making. For example, we could decide that the AI will never move a peer past the border of a discrete trust barrier, in order not to make any security/privacy related mistakes. The users may only drift within the boundaries of the circles, but the user might notice a peer waiting on the rim of a circle and decide to manually drag them over the trust boundary. Another, option could be that the AI would not make any active steps, but rather set up a list of suggestions, that the user could graphically brows and "play out" whenever he finds the time.

- ***Extending the reach of senses*** – Another feature of the social dashboard is its ability to extend our senses to some degree. A person walking down the hallway might get notified that a friend is nearby. The peer's device might be sensed by the radio interface of the user's device, and initiate a notification to the user. Even though the user might not see his friend directly, they would know they are adound and might check behind a closed door, or simply call up the friend and let them know they are close by. Actual sensing distance would depend on the technology used for the radio interface. For example, Bluetooth and Wi-Fi will have different ranges of reach. An ultrasound based interface will have different penetration properties through walls than a Wi-Fi radio.

## 5.3 Generalizing the Social Dashboard

There are many more things that can be done with the Social Dashboard as we advance further. There are multiple degrees of freedom that we are not making use of to the fullest. For example – the angle that the peer entities are presented at, the color of the peers, or the size of the peer icons (which might be used to represent features like importance, or even approximate distance from the user). The look of the dashboard itself does not have to remain in its current radar-like form, which would not work well for devices with too small screens. Another thing not depicted here but already implemented is the option for users to represent themselves and their peers with unique pictures and avatars, and not just the generic icons currently depicted.

Finally, there is a lot of thought given to using the Social Dashboard interface as a general interface to manage one's social network and contacts – for example a way to configure Facebook's friend list permissions and email contacts. For those applications we should assume greater numbers of peer entities that are displayed compared to only active peers in the user's vicinity, and would need to implement better interfaces and filters, as briefly depicted in Figure 5.6.

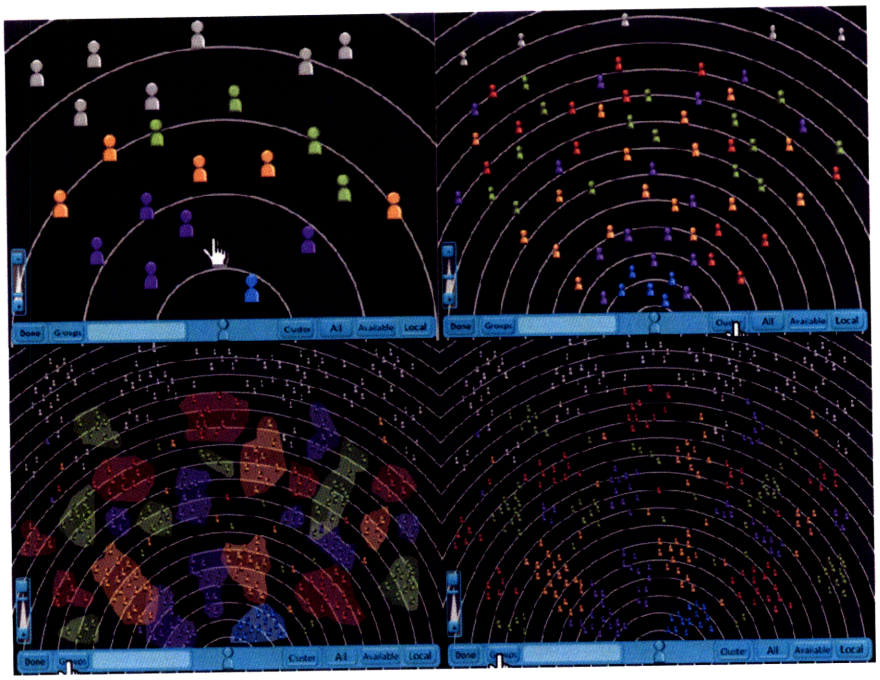



Figure 5.6 - Initial sketches of extending the dashboard to manage all of one's contacts.

# 6 Social Area Networks (SocANs)

*"It is not an exaggeration to say that the future of modern society and the stability of its inner life depend in large part on the maintenance of an equilibrium between the strength of the techniques of communication and the capacity of the individual's own reaction."*

*-Pope Pius XII, 1950*

In this chapter we turn to generalize the ideas and concepts presented and discussed in the earlier chapters. This time we do not limit ourselves to close proximity networking interactions or device-to-device interactions. We focus on the social inspiration of our earlier network designs, and expand it to the general area of networking and digital communications. We place the individual user in the center again, and try to give users the tools to manage and better comprehend their networks. We also propose to use these techniques to eventually help create dynamic network protocols that are "soft" by nature, in a similar way to human-to-human protocols, rather than strictly defined protocols. We motivate this approach, discuss open problems that it could help mitigate, and frame our suggested architecture in the context of existing technologies and architectures – for example how it would integrate with the traditional network layers and where it fits in the evolution of network archetypes.

The main contribution of this chapter is the introduction of "Social Area Networks" (SocANs) - a concept encompassing network architectures built for and around people and their social relationships. In these architectures, the users' social information pervades all the way through the network stack, down to the lower layers of the network, and is used for configuring the network's behavior and its services to the users.

We review existing networking paradigms and architectures as an underpinning to the general overview of SocANs and how they integrate with all layers of the network in section 6.1. We then introduce the high level definition of Social Area Networks and why it might be useful to model distributed network architecture after a human social metaphor (section 6.2. The discussion moves on to motivate our use of social context for configuring network device behavior (section 6.3).In this section, we start by presenting the existing problems of network configuration and management, with emphasis on distributed network architectures. We also touch on the need to give more control to the users over their personal information and content. We then review existing approaches like autonomous configuration agents and game theoretic approaches that use economic incentives. We end this section by discussing our social awareness approach for network management. Section 6.4 discusses the human interaction protocols that inspire the SocAN approach. We then move on to discuss the practical uses of social information for configuring network behavior (section 6.5). In this section we review excising work that uses social information for configuration, and note how existing work usually remains at the application level and does not pervade the network stack. We then give some of our examples for using this information all the way down to the physical link layers of the network. Section 6.6 discusses the other side of SocANs – the ability to use network logs and information for inferring and influencing social behavior of the users. We close the chapter with a summary of features that we would like to project to our network devices.

## 6.1 Background: Network Archetypes

Traditionally, data network types have been organized in a hierarchical manner according to geographical and physical parameters, with Personal Area Networks (PANs) being the smallest and Wide Area Networks (WANs) being the umbrella architecture, with other types such as Local and Metropolitan Area Networks (LANs, MANs) located along this axis.

[Harte, 2003] gives a nice o gives a nice overview of the common types of networks hierarchies that are currently in use. Personal area networks (PANs) are short-range data communications systems that are primarily used to interconnect peripheral equipment (such as a mouse or keyboard) with a local computer or computing system. They are also used in the context of connecting different personal devices, like a user's laptop, personal digital assistant (PDA), and mobile phone. Typically the connection is made by means of Bluetooth [Bluetooth SIG] or infrared (IR)[see IrDA] communication. Local area networks (LANs) are designed to reliably transfer large amounts of data quickly and error-free over a very small area such as an office. Metropolitan area networks (MANs) facilitate LAN-to-LAN information exchange in a local telephone exchange area. The use of a wide area network (WAN) allows for information to be exchanged between LANs located at significant distances from each other. For example a LAN in Chicago sharing information with a LAN Seattle would do so across a WAN.

However, a PAN is an exception to this locality-based hierarchy, as it introduced a more progressive notion: PANs are not about connecting just any arbitrary devices residing a few feet from one another – they add the concept of networking devices being used or even owned by a specific user. Another example of a network type that transcends the physical and geographical boundaries is a storage area network (SAN), which present a view of the network from the storage element perspective. A SAN provides not just a service of connectivity but also other features that are relevant to storage devices, such as dealing with backups and survivability of the stored data. It is along these lines, of networks that are defined by context and use cases, that the concept of "Social Area Networks" emerges.

## 6.2 Definition: Social Area Network

We shall define a Social Area Network, or SocAN, as a term encompassing network architectures built for and around people and their social relationships. In these architectures, the users' social information pervades all the way through the network stack, down to the lower layers of the OSI model [H. Zimmerman, 1980], and is used for configuring the network's behavior and its services to the users. The SocAN's main component can be thought of as a social awareness layer or a "social engine" that could be implemented either as a vertical layer that crosses the boundaries of all existing OSI layers, as seen in Figure 6.1. Analogously, it could also be viewed as a component that is part of the management layer of a network device, which is able to interact with the existing layers and protocols and set their configuration parameters. Note that Figure 6.1 as well as this document's general approach adds an eighth layer at the top of the traditional seven OSI layers, depicted as "End-User Application Layer". OSI's layer seven, "Application", refers to network applications which end users are not necessarily aware of, like File Transfer Protocol (FTP), Domain Name Server (DNS), or Dynamic Host Configuration Protocol (DHCP). The end-

user application layer refers to network applications that the user directly operates, like email clients, instant messengers, or peer-to-peer file sharing applications. These applications could also be configured by the social awareness layer and inform it with updated social parameters of the user. In fact, the connection between this layer and the social awareness component is more straightforward than the other layers, and most of the related work on socially inferred parameter configuration is performed in this realm, as will be discussed in section 6.5.1.The social layer uses the network stack's state and additional information as input, and its socially related output can set parameters and provide additional functionalities as part of the control plane of the device. Sections 6.5 and 6.6 give more concrete examples of this bidirectional interaction.

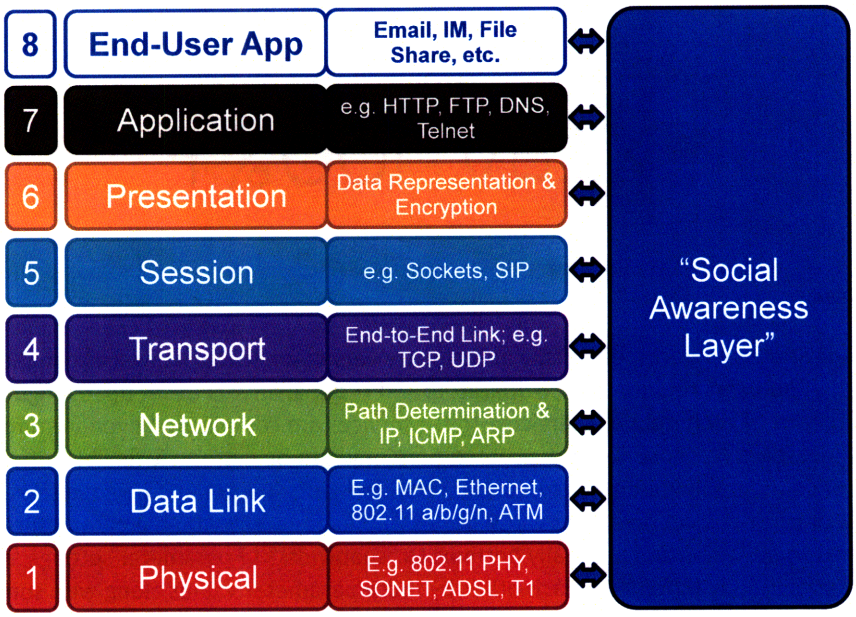



**Figure 6.1 - Examples of the traditional seven OSI model layers, together with the "Social Awareness" vertical layer and the additional "End-User Application" layer, which is not part of the traditional OSI model.**

# *6.3 Motivation: Why social context to configure network behavior?*

## 6.3.1 The Problem

### *Need for Better Network Management*

We desire our networks to be scalable, easy to set up, and simple to configure and sustain (the terms "self-assembling" comes to mind). This is an absolute necessity for distributed systems intended to be maintainable by end users. Nonetheless, this is also quite useful even for network deployment managed by information technology (IT) professionals. Figure 6.2 shows a symbolic depiction of this problem – Bluetooth technology was supposed to be the epitome of plug-and-play networking. The picture, taken during July of 2008, shows a

new service by electronics retailer 'Best Buy' to help people pair their Bluetooth devices.


**Figure 6.2 - July 2008: Best Buy's Bluetooth pairing service (includes testing and demo) [Source: Ricker, 2008]**

Communications networks, especially at scale, are very complicated as well as complex systems[12]. End-users require service operators and IT departments to set up, configure, and maintain most of their network services. These services are complicated even from the experts' perspective: Most network protocols are made to be extremely flexible, offering dozens of configurable parameters like counters, time-outs, packet and payload sizes, bitrates, or error correction settings. TCP, for example, has many variable parameters. What is referred to as "TCP flavors" usually pertains to "snapshots" of parameter settings, and these flavors usually differ from each other by these parameter values along with some minor behavior changes. Such parameters exist throughout the network stack, like TCP's congestion control, Ethernet's maximal transfer unit (MTU) sizes, Wi-Fi's exponential back-off configuration, or the countless Quality of Service (QoS) settings. [Tanenbaum, 2003]

The bottom line is that the aforementioned flexibility is merely a façade, since these values are nearly always set by the developers and service providers to some ***default*** value which is probably good for the common cases, but definitely does not make use of the full capabilities of these protocols under all contexts. Understanding and improving such values is commonly a matter of academic or corporate research, and the source of revenue for an entire market set up to provide products and services for managing large scale networks. The vast majority of networking professionals either makes use of such services, or keeps the default parameter values in place.

The configuration problem described above has different scales of manifestation. Some parameters might affect just a single network device, for example, a wireless device's power management configuration which affects its battery life directly. Some parameters affect a bilateral communication, such as some of the TCP windows size configurations. Others affect groups of devices directly, for example medium access parameters in shared mediums like

[12] For better understand of why "complex" differs from "complicated", see [Anderson, 1972] and [Miller & Page, 2007].

Ethernet or wireless LAN, which include values such as MTU size and and the exponential back-off timers in case of packet collisions. These affect all devices that are in radio range or part of the shared medium, since they cause collisions and blockage on the shared resource. Parameters which have an even larger network effect are those that span the network itself, like routing or link-state protocol parameters. Even though our approach for dealing with the network management problem could improve behavior for single-device issues, the parameters that interest us the most in our discussion are those parameters that affect a collection of devices.

### *Need for Owning and Controlling the Digital Aura*

A recent IDC research study from March 2008 deals with the great increases in the amount of digital information that is generated and stored, and its expected upwards trajectory [Gantz et al., 2008]. Although sponsored by a commercial storage enterprise and thus likely to lean towards overestimating rather than underestimating storage demands, the projected tendency does seem accurate enough, and the study has two relevant observations that are relevant to our discussion. First, it finds that out of the digital footprint that is associated with an individual , "only about half is related to individual actions - taking pictures, making VoIP phone calls, uploading videos to YouTube, downloading digital content, and so on", while the other half is generated by external elements – surveillance photos, Web search histories, financial transaction logs, etc. The second observation is that "*approximately 70% of the digital universe is created by individuals, but enterprises are responsible for the security, privacy, reliability, and compliance of 85%*" [Gantz et al., 2008].

Many times it is desired and necessary to have an expert service provider take care of aspects that the individual does not have the ability or aspiration to perform. However, this dissonance between the creators and viewers of the information to the managers, keepers, transmitters and maintainers of said information can lead to problems for both sides.

In recent years there has been an increased public awareness to privacy issues when dealing with communications service providers. Many events of undesired exposure and use of private information have made it to the press. Some of them were about security vulnerabilities of the service provider, which led an unintentional leak or an intentional hacker attack, for example the hack of TJX's databases which led to the theft of the information of 45.7 million credit cards ["TJX Says Theft of Credit Data Involved 45.7 Million Cards," 2007]. In another case, Facebook had inadvertently exposed the birthdates of its 80 million members [McMillan, 2008].Other events involved the service providers breaching user privacy or user requests with regards to the state of their information. For example, Facebook's controversial launch of its "Beacon" advertising system, which tracks and publishes items bought by Facebook members on outside websites, as mentioned in a [Aspan, 2008]. Another problem mentioned in the article is that Facebook ignores many users' requests to delete their private account data, and instead attempts to save it indefinitely, against the users' wishes. Sometimes users themselves are not aware of the extent of exposure of their digital presence, and the potential implications of its exposure to undesired parties. [Tucker, 2008] gives several examples of trials that were decided against the defendants because they posted incriminating information on their public online social network page. For example, in the trial of a 2006 drunken driver that was involved in an accident where a passenger was killed, the prosecutor was willing to recommend probation for the defendant, until he checked out her MySpace ["MySpace,"] page which featured a

picture of the defendant, taken after the accident, where she was drinking alcohol and joking about it. Please note that this story is only given as an example for undesired exposure of one's digital aura, and we do not mean to imply that any of our proposed solutions to the privacy problem are intended to protect people like the defendant.

The fact that the owners and creators of the information are not the ones responsible for managing it could be undesirable for some enterprises as well, since this might put them in risk of lawsuits for privacy breaches and leaks, and forces them to take responsibility for managing and securing the user's content, as well as scaling up as the amounts of data increase.

These reasons join the motivations described in the previous chapters for the important need to give end-users the control and understanding of their digital aura. Similar issues of control over information are very relevant to discussions of civic media and journalism in general for scenarios where they run counter to the authorities – for example under oppressive regimes who attempt to establish their rule by limiting the free flow of information. Technology and media have evolved enough at this point in time that we can give full control over all of the nested 'onion-peels' of communications media[13]to the individual.

## 6.3.2 Related work: Decentralized Network Management

There are different ways to approach the decentralized network management problem. We overview some of these in the following subsection.

### 6.3.2.1Autonomous 'Intelligent' Agents

Nowadays, machine learning methods are being proposed more and more as methods for solution of complex optimization problems in the field of communication. Here are some examples of proposed solutions for networking problems utilizing machine learning methods. Many of these are centralized, but some are distributed.

We give just a few examples: Geurts and Leduc [Geurts et al., 2004] proposed a machine learning method to improve congestion control over wireless computer networks based on the decision tree boosting method. The algorithm in their paper is used in one of the end systems and can classify and determine the cause of a packet loss only from information available at that end system. The classification of the cause of loss enables to control the congestion in the network (by modifying congestion window size). Boyan and Littman [Boyan & Littman, 1993] proposed reinforcement learning (RL) algorithm for dynamic packet routing through a communication network. The algorithm in which they propose is to use the time takes to deliver a packet as reward and the node to which we should deliver the packet as the action. The state is a function of the current node, that the packet is destined to and the neighbor node. Brown [Brown, 2001] had proposed an algorithm for packet switching using RL where the goal is to find the optimal contention arbitration policy. In his algorithm the reward is determined by the size of queues in the stations. Ruiz et al. investigated using a distributed genetic algorithm agent for learning to decide when to

[13] For an overview of McLhuhan's discussion of media within media, see Appendix.

trigger quality of service (QoS) adaptation in a way that attempts to optimize for better user-perceived QoS (rather than absolute technical measurements) [Ruiz et al., 2004]. Koch and Westphall [Koch & Westphall, 2001] discuss how Centralized approaches to Network Management have demonstrated inadequacy for efficient management of large and heterogeneous computer networks. They present a general approach and their own architecture for using distributed artificial intelligence (AI) agents for network management. However, demonstrating a working system is out of scope for their paper.

The key point here is that most of these methods do not make it out of the realm of research. This is due to various reasons, from computational power to the obvious fact that real life is never as neat as lab conditions and many of these algorithms fail to perform well in arbitrary conditions. Even when they do, these types of algorithms usually configures just one or very few parameters, and do not perform full network configuration. Many times, by focusing on solving one problem, they create another.

#### 6.3.2.2 Economic Incentives Approach

One of the more recent approaches in designing distributed systems is to use "economic incentives" and draw on economic and game theoretic principles. This could be considered a state of the art in distributed network architectures. Related work has been performed in this field with regards to wireless mesh and peer to peer (p2p) systems [Lai et al., 2003; Ma et al., 2004]. Bittorrent is an implemented and wide spread system that uses these principles [Cohen, 2003]. Current designs of such systems treat all nodes in the network in the same manner. These systems implement accounting mechanisms that give credits for services provided, or exchange actual micro-payments. However, when considering real-life relationships there is differentiation between the peers. Most people probably don't need or even want to do "tit-for-tat" accounting for sharing files with family and friends or routing mesh packets for them. Another example from a business scenario – in a conference mesh network - do I really want to route my data through my competitor's machine? If we look at the communication network as an augmentation of our social interface to other people, shouldn't it also represent our social affinities?

Another thing to note about people is that their incentive systems are dynamic and not constant. In the same way we act differently when we are at work, at home, on the road or on vacation, perhaps our network incentives could change as well. One of the improvements proposed by the SocAN approach is to augment such economic incentive models with socially inspired weights, like trust or types of relationships, which will be dynamically modified according to context.

### 6.3.3 The social awareness approach for network management and interaction

As briefly mentioned above, our approach, embodied in Social Area Networking, is to use concepts from human social interaction for network configuration. Oviatt, in her study of human centered design and cognitive load theory, touches on the importance of leveraging from users' experience, knowledge, and engrained behavioral patterns as well as accommodating user's existing familiar work practice rather than attempting to change it [Oviatt, 2006].

These principles are directly relevant to our goal. Network mediated communication, whether the end points are human-to-human, human-to-machine, or machine-to-machine, is usually associated with social communication metaphors. For example the notions of "talking", or passing messages between parties. In SocANs we extend this metaphor to the management plane as well. This manifests in two levels:

- *End user interface* – Since we want users to be the administrators of their own devices, we want to provide them with network interfaces and management tools that leverage their engrained social knowledge.

- *Developers* – We want to create back-end mechanisms and definitions that would allow translation between social terms and social context to 'hard' engineering terms, which can be implemented and utilized with existing device and protocol parameters.

By doing so we present the users, as well as network developers, with a *vivid* model that they could relate to and utilize in order to take full ownership and control over their networks. We would like the network's management as well as its regular interaction interfaces to be comprehensible and simple for the end-users, so that they could unleash the full potential of the network as a platform for advancing social collaborations and efficient use of information.

The idea of Social Area Networks proposes a middle ground of sorts, as it aims to provide a framework for applying different sets of configuration parameters according to the current social context. This approach is by far not claiming to provide optimal dynamic configurations, but there is good reason to believe that it will lead to better performance over the default situation, simply because there is not just one single default value, but the option to provide multiple values to pick from in each situation. This approach intends to augment and enhance existing solutions, rather than supplant. Also – we should remember that the user's goal is not necessarily mathematical optimality.

The network's goal is the owner's goal – whether it is a single user with her personal devices, or a large corporation with scores of devices. An example to illustrate is if a company was able to simply set up its network QoS configuration and security parameters according to a company's hierarchical structure and desired connections between different departments and employees. This is done manually today by IT teams, as they manually set parameters that would, for example, make sure the CEO's network connection will have precedence over others, or making sure the customer facing departments will have continuous network service. Perhaps we could define a set of simple rules to automate some of this social knowledge and 'networking-commonsense-knowledge', and translate different relational links into network behavior configuration.

In subsequent sections of this chapter we try to better justify our approach with concrete examples. In order to implement these ideas, we must first attempt to understand how humans react, manage, and interact with their counterparts in real-life. We already touch on this in our theoretical background review in chapter 4. In this thesis we present just one instance of a possible implementation. Effective representations of social contexts for network management might be implemented in many different ways, and this is an interesting research question on its own.

## 6.4 Inspiration: Human Interaction

Human societies are distributed, multi-agent systems. In these systems each individual is also a node in a (social) network or, to be more accurate, in multiple networks to be more accurate. Each person possesses a set of skills allowing him or her to adjust to dynamic network conditions and ever-changing contexts. When conversing over an identical topic in different situations, people would act differently according to the situation, the context, the identity of their counterpart(s) and even according to other people that are nearby. They use different wording, different gestures, or in networking terms - different communication protocols. In our social world, if someone steps out of line – society has protocols to correct them or put them in their place. Society rewards those who act benevolently and penalizes selfish behavior. Moreover, no matter whether one acts especially "good" or especially "bad" – the word would usually get around and affect how society treats that person. Researchers such as Goffman [Goffman, 1959], Barker [Barker, 1978], and Wright [Wright & Barker, 1959] studied these human social behavior patterns. We reviewed some of the relevant work of these and others as part of our VPM discussion, in section 4.1.

As mentioned, social context might include parameters like the social setting, or the number and identity of peers who are part of the conversation, as well as of peers who are not part of the conversation but may overhear it. As an exercise to the reader, let us look at how humans interact with each other. Think about the different social interaction protocol parameters - like word selection, tone of voice, volume, handling interruptions, etc. – that you might employ in the following situations:

- One-on-one conversations: With a friend; With one's supervisor, vs. one's peer, vs. one's subordinate; In a noisy environment vs. in a quiet environment.
- Communicating with a few people: In a group/team meeting; During a business meeting; During family dinner (in which the protocol is family-dependent...); When engaged in remote interaction (such as a teleconference).
- Communicating with many people: Giving a lecture; Presenting at a shareholders meeting; At the airport, when the ground crew informs the waiting passengers that flight is overbooked and they will seat passengers by first-come-first-served.

There are many parameters that might run through our mind during each of these situations: Is this an open discussion? Should I go first? Should I go last? How aggressive should I be in trying to get the right to speak? Should I whisper? Should I shout? Can I interrupt the current speaker? Is there a moderator? Can I get emotional? Do I have to answer this question?

There are usually no strict rules of nature as to how we are supposed to interact with our peers. Instead there are social norms[14]. Goffman would discuss these interactions with respect to "roles" and theatrical-like "performances" [Goffman, 1959], Barker would call these interactions "executing a program" which is part of a behavior setting [Barker, 1978], and Meirowitz would probably use terminology of different audiences and categorization [Meyrowitz, 1985]. Another aspect to think about is that in social life there are always exceptions - Arguments with family members, for example, are different than those with strangers. Society has these sorts of "soft" protocols, and humans have an ability to detect social context and modify their behavior. When people do not do this, due to ignorance or a

[14] Which might sometimes seem like hard-coded rules of nature...

medical condition like Aspeger's syndrome, they seem odd and out of place to the rest of society[15].

The question we ask here is whether our devices could ever act in a similar manner to well functioning members of society. This work's premise, of course, is to answer this question positively. The majority of this document deals with ***manually*** establishing these configurations – setting the correlation between different social contexts to device behaviors. However, in the conclusion of this document where we mention future directions, we touch on the possibility that based on the lessons learned through this work and likeminded research, one day our devices might be able to autonomously discover these behavior rules. Before getting ahead of ourselves, let us go over some more concrete network behavior examples and related work on the matter.

## *6.5 Use of Social Information for Network Configuration*

### 6.5.1 Related Work: Use of Social Information for Configuration

Pentland at al.[Pentland et al., 2006] build on the results from reality mining [Eagle & Pentland, 2006] and propose to use the inferred social data in order to aid in setting privacy, sharing, and interest settings. In the context of this thesis, these settings would be at the top most layer of the application stack – at a level that the user is aware of. The work presented here attempts to take similar social information, but make use of it *throughout the network stack*, down to the lower layers. A clue that this may be a useful approach can be seen in [Miklas et al., 2007], where the authors used the reality mining database in order to simulate several network scenarios and show that DTN routing protocols, simple firewalls preventing a worm infection, and a mobile P2P file-sharing system can benefit from exploiting social information. Others have also suggested using a social network to inform routing decisions for DTNs – delay tolerant networks [for example Ford et al., 2006; Marti et al., 2004].

Aside from routing, nearly all of the existing works that use social information remain at the user and application level. Recently several startup companies are attempting to create search engines who's results are influenced by one's social network ["Nsyght,"], ["Delver,"]. Other initiatives use social information for inferring privacy and content sharing settings. The Mob-Media project is deals with learning media sharing and social interaction information for use in areas of media distribution and privacy settings [Madan, 2008]. We also discussed more socially aware applications in section 2.4, where we talked about Face to Face networking applications. Finally, there are several large scale initiatives aimed at tagging socially related information on the World Wide Web and online networking sites – Such as Facebook's API's [Facebook], Google's Open Social ["Google Open Social,"], the Data Portability initiative ["dataportability.org,"]. These initiatives allow developers to build socially aware applications as well as move data in and out of social information repositories, like online social networking sites. This has made the socially aware application area "blow

[15] Sometimes it might seem that the ratio of people suffering from these afflictions in the academic world is higher than their portion in the general population.

up", but still, most of this information is used at the user application level and does not go below it.

### 6.5.2 Additional Examples

To the best of the author's knowledge, aside for the routing example, there has not been previous work that suggested using social network information and applying it for device behavior configuration in the lower OSI layers. Even more than specific application ideas, it is important to define an interface model that would connect the lower layers to this information, and open up this wide area for further innovation. For this reason, in this section we shall give some examples of how social information can be used for configuration, for each of the OSI model's layers.

We have already mentioned the possibility of modifying routing information according to social network information. That would be part of the OSI's Network layer.

In the Physical layer, for example, we could integrate SocANs with the ongoing work on cognitive and software defined radios [Fette, 2006; Mitola, 2006]. These radios can coordinate their own protocol, or frequency hopping sequence. If a cognitive radio would be aware of the relationships of its owner, for example be able to recognize the devices of the owner's family members, these devices could coordinate a frequency hopping sequence of their own, helping performance and security.

In the Data Link layer, SocAN approach could configure medium access parameters. For example, a Wi-Fi has configuration parameters for medium access attempts – timers and counters that configure how it should act after a packet collides, for example – how long to wait before trying to retransmit the packet, or how many times to try before giving up. These parameters are usually left at a constant default. A company's devices could use the company's hierarchy or other social preference rules to dynamically change these parameters according to sensed devices. For example, an employee's laptop may be more "polite" when the boss's device is around, giving the boss a better chance to send his data. Alternatively, a company server or printer may get these privileges from nearby personal devices, since their activity gains precedence. In a similar way we could play with 'politeness' and 'aggressiveness' of similar parameters in other contexts. Another example at the Data Link layer deals with authentication. Currently, only one type of authentication is usually implemented by 'authenticating parties' – for example a Wi-Fi access point. Perhaps we could use the social information in order pick an authentication method out of a set of multiple methods. For example, do a different authentication for familiar peers vs. strangers. Familiar peers, or members of an organization, might have a quicker authentication process, whereas strangers might be diverted to an authentication process that is more secure and demanding, and perhaps even force them to expose themselves in real life (e.g. –" your has been generated and is waiting at the front desk, please go there to pick it up").

The Presentation layer deals with aspects like encryption. Another corporate example is that the device would automatically turn on encryption when communicating with a co-worker, but not use it when interacting with devices that belong to friends or family. Sensing 'stranger' devices in the vicinity might cause a family's device to encrypt information.

A multi-layer application could be an advanced "social firewall". [Miklas et al., 2007] suggested that a device could use the differentiation between friends and strangers to repel

digital 'worm' attacks. A much more complex version of a socially aware firewall could be devised, that would configure parameters in multiple network layers, as well as react to many kinds of social relationships and groupings (and also provide protection on the border between these social groupings).

## 6.6 Use of Network Information for Social Inference (and Influence)

Previous work, particularly work performed at the Media Lab's Human Dynamics research group, has shown how social sensing can be performed through mobile wearable devices, and how through offline analysis we can reliably recognize social patterns in daily user activity [Eagle & Pentland, 2005, 2006; Olguín, 2007; Olguin & Pentland, 2007]. Particularly relevant is the reality mining project, which used a mobile phone application to collect data from more than a hundred people over several months. A particular thing to note about that project is that all data was concentrated at a centralized server and analyzed offline. One of the ideas of our own work is to extend from these methods, and try to develop a platform that would allow performing the reality mining in an automated manner on the user devices themselves. The hypothesis is that we don't need the entire network picture in order to map out a single person's relationships. Much of these social data collection, performed though network devices, has been done in the wireless and face-to-face realm. We review more of these technologies in the chapter focusing on face-to-face networking, in section 2.4.3.

Sensed social information can be used to build a user's social network and imported into social networking sites. Instead of manually adding a person's friends and other relationships, and configuring affinity groups and so on, we could use learned information to infer a lot of this information and import it to the different sites.

A large part of these experiments has been performed with specialized hardware (like sensor 'badges') [Olguín, 2007; Olguin & Pentland, 2007] or preconfigured mobile devices that were given to participants. This means that most of this data was collected in closed contexts and for a limited time – in campus or corporate environments, and in experiments of predefined length. One of the goals of this thesis is to use SocAN based applications in order to perform social experiments that people can download and install on their existing devices, and use in their everyday lives for very long terms (with user permission, of course). This could lead to social data collection at unprecedented scale, which will hopefully help advance the burgeoning field of *computational social science*.

## 6.7 Discussion

Hopefully we have shown the potential of the Social Area Networking approach and justified the need to define SocANs as their own type of networks that might merit standardization and could push network and communication device vendors to provide more capabilities and access for collecting information and setting parameters in the devices. We also tried to show the bi-directional benefit of this approach – both to device performance and management, and to user's social interactions.

Following we have aggregated our initial list of the aspects of the human social realm that we want to ***project*** to our devices:

- Identities – multiple identities, real identities, temporary aliases.
- Groups and affinities, and our ability to signal them (as trivially as wearing a t-shirt that states that affinity).
- We have a map of relationships with individuals as well as groups in our lives. These relationshipps are very heterogeneous.
- We have different levels of trust associated with our relationships, and this affects how we behave with different peers.
- There is an important separation between things that are private in our minds, and things that we do. We might choose not to present what we privately think, and we might choose to present a different message to the outside.
- Awareness of our surrounding – different senses, inference on what we see even if we are only bystanders to a social interaction.
- Memory – we remember interactions and past behavior, even of strangers.
- Easy communication with strangers and anyone around us without having to use complicated rendezvous and protocols.
- We don't need a full social network picture in order to function in a social system. Our own views are usually sufficient (and many times we have no other alternative).
- We try to control our presence and appearance – we can choose not to answer a ringing telephone. We can choose not to reply immediately to every email.

We already visited many of these points in the earlier chapters dealing with the VPM framework and the Social Dashboard interface. Hopefully we have shown in this chapter the potential to implement them in many other realms of networking other than face-to-face interactions. These aspects can be used both as inspiration for implementing functionality, and also as a way to understand the needs of the end users. In the next chapter we go into describing the Comm.unity software package and architecture – that already implements many of these ideas, and can serve as a platform for developing and testing new ones. All of these aspects are part of what makes us who we are, and define what others see of us. The digital version of these is the makeup of our ***digital aura***. It is ours, it is always with us, and we need to reclaim the control over it.

# 7 The Comm.unity Platform

*"Nothing is built on stone; all is built in sand. But we must build as if the sand were stone."*

*-Jorge Luis Borges*

The Comm.unity platform is an open source middleware and API for socially aware and/or Face-to-Face Networking applications. It is aimed at both application developers and researchers alike.

Comm.unity is designed as a platform for easy development and deployment of a variety of socially informed applications for sharing information and content among peers who are in close vicinity of one another. It includes mechanisms for using the mobile device as a sensor for collecting social data as well as for learning of the user's social relationships. It also offers a socially focused user interface framework that allows the users to interact with people and devices around them, as well as very easily understand and modify the configuration of their own socially-aware device. Aside from the stated principles of the FtFN and SocAN frameworks, Comm.unity implements a delay tolerant network (DTN) [see Cerf et al., 2007] architecture. For developers, it offers the ability to quickly create distributed wireless applications over a vast set of devices and network interfaces, without the need for deep network programming knowledge. Comm.unity's extensive logging functionality as well as its interfaces to various sensing modalities could be employed by researchers to catalyze the deployment of social experiments and data collection using mobile devices.

As its name may hint, aside from the social and "community" connotation, the platform aims to create a "communications unity" between network devices. The idea is to enable as many different types and models of devices to communicate with each other, empowering the users and diversifying communications networks. Comm.unity makes use of the existing network interfaces and radios that a device might have. It can currently utilize 802.11 (WiFi), Bluetooth, and any interface that uses standard TCP or UDP network sockets, such as a wired Ethernet interface card. It defines an application programming interface (API) for adding new types of radios, from commercial ones to self fabricated (like Internet0 [see Krikorian & Gershenfeld, 2004]). Attention is given to devices that have no screen or small screens, by designing the user-interface library of Comm.unity with the ability to be physically detached from the core of the package. It is able to connect remotely, through the network, to a device running the core engine. This chapter assumes some basic knowledge of network terms, technologies, and standards for background information regarding these terms see [Tanenbaum, 2003].


Figure 7.1 - Comm.unity Platform Logo

## 7.1 Comm.unity Design Goals

The high level goals of Comm.unity are, well – high level. In general, Comm.unity is intended to implement all of the individual features and requirements that were laid out throughout the document. Some modules implement the FtFN principles, some implement the SocAN principles and features and the Comm.unity platform also hosts our implementation of the VPM architecture.

Following are some of the ***engineering goals*** of Comm.unity:

- **Unifying FtF networking technologies:** As mentioned in chapter 2, there are many functions common to most if not all of the FtF networking technologies. All involve basic services such as discovering nearby peers, exchanging messages, or transferring streams of content, for example. All require parameter configurations, like physical layer addresses. They all have to deal with dynamic topologies, where peer devices come in and out of contact as the radio environment changes or as users move about.

  In addition to this, the ways to programmatically configure the physical interfaces may differ between operating systems (Windows, Macintosh, and different flavors of Linux), or even between different models of devices supporting the same networking standard. For example, some wireless interface devices and drivers cannot perform scanning while associated to a network, and some cannot be switched to a different network without being disabled beforehand. Comm.unity offers a framework to add detection and treatment of specific interface nuances.

  There are existing packages that simplify dealing with some of these issues, for example the MadWifi wireless driver ["MadWifi,"], or the LightBlue Bluetooth library [Lam, 2007]. However, these libraries usually offer a limited set of features that does not cover all of a developer's needs, and we have not seen any that abstracts multiple interface types, as the FtF framework suggests. The idea in Comm.unity is not to duplicate the functionality but rather to create an abstracted interface to these packages.

  Even with the existence of such packages, many application developers would rather focus on the application itself and not deal directly with the network drivers and APIs. Nowadays, many application developers either assume that parameters are set externally (for example, having the user manually connect to an ad-hoc or infrastructure network, or setting an Ethernet device's IP address automatically through a DHCP service). These assumptions do not work well when dealing with unmanaged distributed scenarios and high user mobility, like in our model of FtF networks.

  To summarize: *The goal of Comm.unity, in this respect, is to provide a unifying abstraction layer over the different interface models and types, which would allow developers to easily maintain and make use of the available network capabilities.*

From the user applications' side (as well as their developers'), we can identify the following options for interaction with the network interface that Comm.unity could offer:

- *Interface specific*: provide the application/developer with a list of available interfaces and let them easily specify which one to use (e.g. WiFi or Bluetooth), and also which way (e.g. broadcast to all, or send to a specific peer or peer group).
- *Interface agnostic*: Some applications do not care which way to get the information across peer devices, they just want it to get there. Any medium would do, *whether* it is WiFi, Bluetooth, ZigBee, or even an infrastructure network.
- *Interface by policy*: Allow developers to provide some guidelines to Comm.unity regarding how to select the interface to be used. For example, try to preserve energy, or try to maximize range when multiple interfaces are applicable. This interface is still not fully implemented would be relatively easy to implement and enhance with new types of policies.

- **Portable/multi-platform implementation**: Continuing the goal of working with different types of interfaces and their nuances over different devices and operating systems (O/Ss), would like Comm.unity, and especially applications built over Comm.unity, to be easily portable to different computing platforms. One of the things we would like to avoid, for example, is time-costly cross compilations and porting of code between different platforms.

  This is the main reason that most of Comm.unity's codebase is written in Python, which is highly portable between different devices and operating systems. Most of the Comm.unity code is pure python and will execute seamlessly over different platforms. The main aspects that require adjustments per O/S are the parts that read or modify aspects in the device's hardware, such as network interface configuration, or parts that deal with the file system, such as the location of the user's home directory or different O/S naming conventions. Even though these aspects are O/S dependent, the goal is to deal with them during the development stages of Comm.unity, so that they too will become transparent to the developers working with the platform.

  In addition, because the platform is designed to run on pocket devices with relatively weak processors, little or no effort is required to have it operate on stronger machines like laptops and desktops. Developers who would like to add new functionality to Comm.unity, or build multi-platform applications would also need to make sure their code is efficient enough to run on the least powerful Group entity icons in implemented prototype.devices on which they aim to execute their applications.

- **Extensible architecture - modular building blocks**: Comm.unity is implemented using modular building blocks, and defines different APIs for passing messages and commands between the different Comm.unity layers and modules. This allows us as well as other developers to add new modules and functionalities in the future. This is similar to the OSI's layered network model [H. Zimmerman, 1980], however the Comm.unity blocks and API are focused on our own goals implementing SocAN and FtF functionality.

For example, we may think of a command primitive for transmitting a file reliably to all nearby peers. Different applications might call this primitive, agnostic to the actual network interface that will be used. Let us assume the primitive function call is "ReliableFileBroadcast(<file_name>)". Each supported interface would implement this differently. For example, an 802.11 interface version might implement a loop that would divide the file into single UDP packet chunks and broadcast each of them individually until acknowledged by all devices in the vicinity. Another way to implement this could be to use a TCP stream to send the file by unicast to each of the surrounding peers one by one, which emulates broadcasting it to all of them. A Bluetooth implementation might use a similar strategy using OBEX file transfer [Bluetooth SIG], since Bluetooth does not implement a conventional data broadcast primitive. A different future implementation might use network coding schemes [Fragouli et al., 2007] for implementing the reliable broadcast, which will make more efficient use of the network under some conditions. Another future extension might implement a virtual shared medium by running Comm.unity on top of a logical overlay network, like Resilient Overlay Networks (RON) [Andersen et al., 2001], or a logical mesh network, as defined by 802.11s [Hiertz et al., 2007]. In this case the "broadcast" protocol would be implemented in a completely different way. Through all of these example, the user application simply implements a call to the "ReliableFileBroadcast" function.

- **Modular runtime**: Another key benefit of the modular architecture is that it allows us to only load the specific modules that are relevant and applicable to the current device or application that is being executed. For example, there is no need to load Bluetooth or camera support if the device does not have it or the application does not make use of it.

## *7.2 High Level Architecture*

Comm.unity is currently implemented as a combination of Python and Adobe Flash code. As summarized in Figure 7.2. As we detail more in section 7.3, the Python side implements the bulk of the platform. Although it is dubbed "back-end", it should be noted that the Python side could run as standalone, outputting to a terminal, a log file, or not at all. For example, one might think of running Comm.unity on screen-less devices, like sensor network nodes, computing embedded in the environment, or a device that people carry in their pockets, such as the Living The Future project's "Amulet" device [Reed, 2006, unpublished]. There are many deployment scenarios where user interaction is not needed and the focus is on network functionality.

When a graphical user interface *is* needed, however, we have found that Adobe Flash provides a very good solution for several reasons, mainly its level of proliferation [Adobe], its portability between platforms, its built in Graphical User Interface (GUI) libraries which make it very easy to do rapid interface development, and its ability to provide the "look and feel" that today's users have come to expect from graphical applications.

Another aspect related to the goals of making Comm.unity portable and robust relates to the graphical user interfaces (GUIs) supported by the platform.

Another key feature that Comm.unity provides is the ability to separate the back-end and the fron-end, and have each run on a different device. For example, a user might run the

system on a mobile phone, but use his PC to access the GUI and configure the device, or a more elaborate scenario where an Amulet's graphical interface is executed from a screen and computer that is part of the environment as the user stands in front of them.

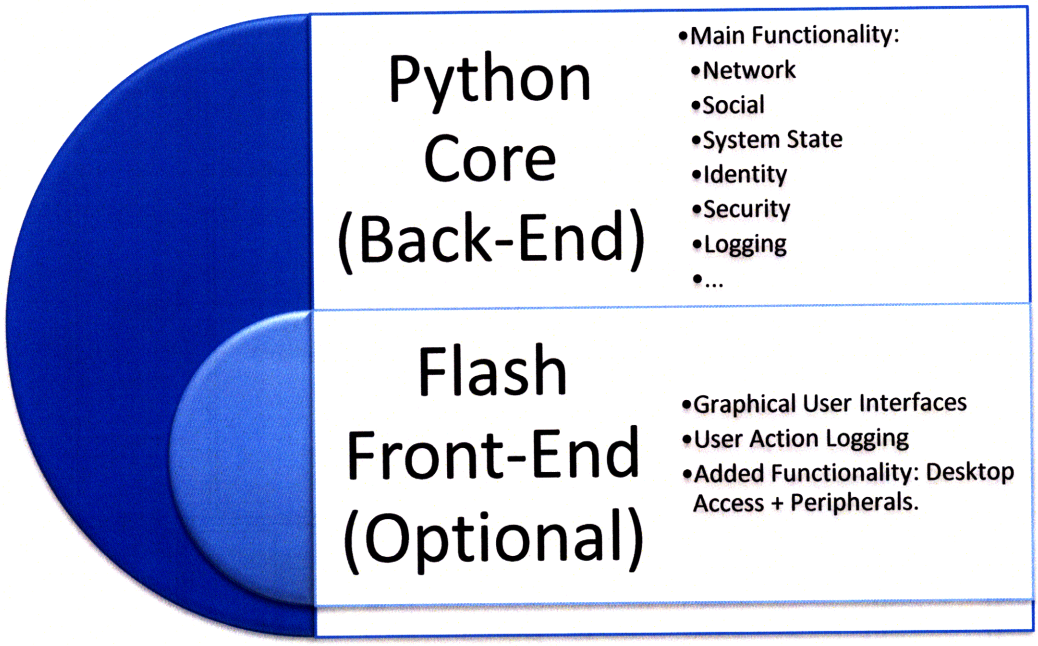



Figure 7.2 - Comm.unity top level

The Comm.unity networking core makes use of an asynchronous networking paradigm, allowing a great number of socket connections using only one process thread – making the program more efficient as far as memory and processing overhead, and better suitable for resource-limited platforms, like mobile phones and PDAs. It is built on top of Twisted, an open source event-driven networking engine written in Python [Fettig, 2005].

## 7.3 Conceptual Architecture Components

Figure 7.3 shows the conceptual system architecture of Comm.unity and its main system components. These components are all implemented in the Python core however some of them might be delegated to the Flash side when it makes sense. This could be especially relevant for multimedia peripherals. For example, Flash Light has functionality that allows very easy connection to a mobile phone's camera and microphone.

The component stack is composed of distinct layers, each targeted at handling a specific type of functionality, such as interfacing to the physical layers, or providing security mechanisms. Within each layer, more specific modules are implemented, which actually perform the actions. Each layer exposes an API of functions and variables that can be used to call on different modules and module functionality, or access state variables. Developers can add their own modules and extend the functionality of the platform. For correct implementations, all modules within a layer must support its standard API functionality.

Note that in this system engineering view, the modules are defined and grouped by functionality. In the implemented architecture, a different module and file structure is defined for the Comm.unity source code, mostly for reasons of code readability and efficiency. Detailing the full file structures is out of scope for this document. In the remainder of this section we will go through each of the component layers, roughly in a bottom-to-top order, and try to give some overview of their functionality.

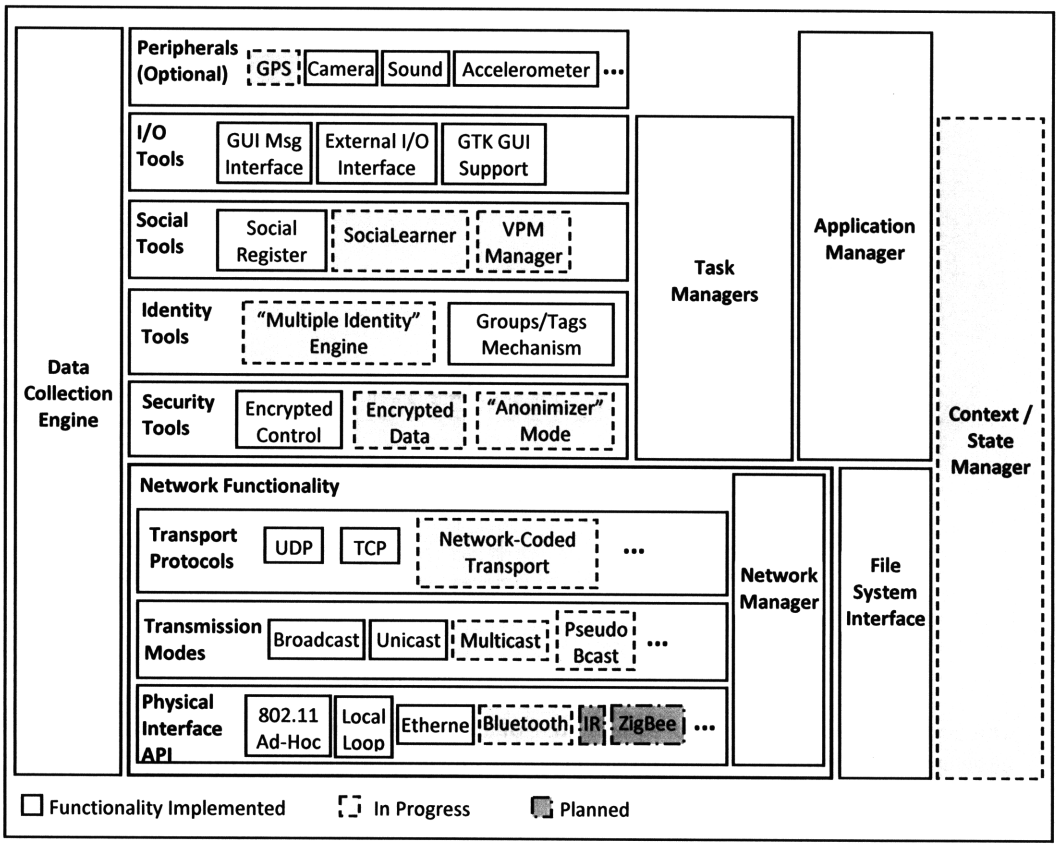



Figure 7.3 - Comm.unity architecture and system blocks (Python side)

## 7.3.1 Network Functionality Multi-Layer

The network functionality layer is actually composed of several sub-layers:

1. Physical Interface API layer, which handles connection to the device's network drivers and hardware.
2. Transmission Modes layer, which hosts modules that define different ways of sending information to the physical interfaces, and
3. Transport Protocols layer, which implements or wraps existing transport protocols for handling different network transmission tasks.
4. Network Manager Layer – this layer implements elements to synchronize and orchestrate between modules in the three previous layers, as well as general network functionality of the device.

These layers are very tightly coupled, and many times we find that new network functionality has to be implemented with a module in each of these layers. For example, adding support for Bluetooth advertisement of Comm.unity applications and services requires logic of how to make use of existing Bluetooth functionality (like it's built in service advertisement and discovery) as well as a need to connect to the Bluetooth device and configure its settings. However, different modules could be reused by different functionalities, for example, once we know how to do a reliable UDP unicast transmission, different transport modules can make use of it for their own tasks – from management messages like advertisements and requests, to user chat messages.

### 7.3.1.1 Physical Interface API

This layer defines the integration of Comm.unity with the specific radio interfaces, operating system and the hardware of the specific device Comm.unity runs on.

This layer's functionality includes:

- Get and set IP addresses of the device, and other TCP/IP network parameters (subnet, mask, etc.)
- Interaction with device's 802.11/WLAN interface:
    - *Get and Set WLAN configuration (channel, network ID, MAC address, ad-hoc or infrastructure mode, etc.)*
    - *Enable/Disable the device*
    - Perform wireless scan for nearby networks
- Interaction with device's Bluetooth interface: functionality similar to the WLAN.In a similar way, modules for interaction with other radio interfaces for example : Infrared (IR) [IrDA], ZigBee [Zigbee Alliance], Internet Zero [Krikorian & Gershenfeld, 2004] or even FM radio as the physical interface.
- Local-loop interface – use network device's standardized local-loop option (e.g. address 127.0.0.1) for debugging or for running the system in a standalone mode or for running multiple virtual instances of Comm.unity on the same device.
- Deal with any needed ARP table or IP routing table get/set functionality (currently not needed).
- Module that makes use of a cellular data link (EDGE, 3G), if available, to bridge between the local FtF network and the infrastructure network (the Internet)

An important key to note is that for the most part, each OS or device platform exposes different ways for interaction with its physical network interfaces and drivers. This means that each supported interface would need special treatment from any device, implemented as a separate module or as a "parent" module that can load code to handle each of the supported cases. This is similar to the way that the Twisted networking package deals with the differences between Windows, Mac, and Linux platforms. Out of all Comm.unity layers, this is the least portable one, and its defined API would be least consistent across modules and interfaces, simply because those interfaces are not fully compatible. For example - Not all allow scan of neighboring devices.

### 7.3.1.2 Transportation Modes

This layer implements modules that define various ways to transmit data for each of the supported physical interfaces. It exposes a set of primitives to the higher Comm.unity layers,

such as "Broadcast", "Unicast", and "Multicast".Its modules implement this functionality in different ways. Here are examples of different modules that implement a broadcast-like functionality:

- Standard 802.11 broadcast message to a predefined broadcast address. It is very simple to use, but the standard broadcast definition comes with a set of "features" that we may sometimes want to bypass using other implementations. For example, broadcast messages have no link-layer acknowledgments, and they are sent at the slowest possible line-rate (in 802.11g, broadcast is sent in 2Mbps bitrate compared to a maximal supported rate of 54Mbps)

This is one of the reasons we might want to use other ways that emulate the broadcast functionality, for example:

- Emulated broadcast by unicast - send unicast the content to each of the surrounding nodes, with a faster transmission speed and possibly using link-layer acks. Cons – might require us to know exactly which devices are around, something that is not always possible.
- "Pseudo broadcast" – improvement on emulating broadcast by unicast – we unicast the content to one specific node, but allow other nodes to "overhear" and keep any relevant information. This reduces the need to send each node the entire content, but only to "make up" for the pieces that are missing and were not overheard [as mentioned in Katti et al., 2006].
- Emulated broadcast over an overlay network – similar to the implementation of 802.11s [Hiertz et al., 2007] or the RON project [Andersen et al., 2001].

This layer exposes both the specific module functions, to allow a higher layer to choose which transmission functions to explicitly call, and also the more aggregated primitives mentioned above. These would have to be initialized to one of their module instances, depending on the desired interfaces, or to some other modules that implement a union of different functions (for example a module that implements broadcast over all available interfaces at the same time).

## 7.3.1.3Transport Protocols

This layer deals with higher level functionality of the networking stack – for example making sure a file is transferred successfully from end-to-end whether it was sent to a single destination or multiple ones.

This layer already implements networking modules such as the following:

- Advertisement broadcasting protocol – periodically send out a message with information such as the identity of the device/user, some information about the groups they belong to, files they have, or anything that application developers would like to implement.
- Request sender protocol: Send out a poll/request persistently until an ack is received or a timeout occurs which will cause it to be marked as failed.
- Request and advertisement receiver protocols: Handle received advertisements and requests (for example, check for CRC correctness and reply with an acknowledgement)

- TCP based file sender and receiver.
- UDP file sender and receiver
- Single message sender/receiver with or without acknowledgements

The above are implemented, as well as several other modules not mentioned. Future modules might be variations on the ones described above, for example implementing a TCP file sender/receiver that can deal with partial file transfers, and resume a file transfer from different peers. There are also other types of modules in development, for example a protocol that looks for network-coding opportunities and uses network-coding to combine multiple packets in order to reduce the number of bits that need to be transmitted.

#### 7.3.1.4Network Manager Layer

The network manager layer implements elements that synchronize and orchestrate between modules in the three previous layers, as well as general network functionality of the device.

Some examples of functionality that would fit into the network manager layer:

- The ability to use information about the state of the network and the peers that are around to select different transmission modules – for example, choose the optimal broadcast module based on the number of peers around and network congestion information.
- Another example would be that if the "social state" of the system indicates that many stranger or adversary devices are detected nearby, it might decide to enforce encrypted transfer of information to a co-worker's device.
- Yet another example could be a collaborative request handling module, where several peers in the vicinity are able to coordinate who will reply to a given request based on different parameters like fairness, battery life, or social relationships with the requesting device...

### 7.3.2 File System Interface

This component handles the file system interface to the file system of the device, and manages the Comm.unity user files (for example user information, application content, and system logs and metadata). Luckily, Python and some of the Python libraries that are in use already handle the many of the file-system nuances of the three major operating systems (Windows, Mac, and Linux), but this layer also has to keep track of Comm.unity's different paths for each operating system, like the user's root directory and different paths for application content. In Linux, for example, the user's Comm.unity content directory is part of the Linux home path (usually marked as "~" or "/home/<username>", whereas in Windows XP it is customary to put application related data in "~\Application Data\" or "C:\Documents and Settings\<username>\Application Data\".

Some of its modules include:

- A set of utility functions that can scan group directories and mark which files are relevant for each group and application, and imports this information to the relevant applications.
- A set of functions to add Comm.unity metadata to any file in the user directory structure – for example calculate a hash to uniquely identify the file, add

information about its creator, creation times, and different events that happen to it as it goes through the system. For example some applications might want to log the path that a file traverses in order to visualize it or give some other service to the user.

- Functions to load and save logs and state of the system.
- Functions to handle a received file and notify the application manager that will in turn notify the relevant application of the arrival of new content. For example – notify a media player that a new song, image, or video has arrived from the network.

### 7.3.3 Security Tools

As its name implies, the Security Tools component houses modules that try to deal with security issues as much as possible, since there are no fail proof solutions. For a more detailed discussion of the security aspects, please see the literature review and discussion in the context of VPMs, in section 4.5. This module houses the implementation of the different VPM security functions.

### 7.3.4 Identity Tools

The identity tools module deals with the different entities (VPM or otherwise) that belong to the host – this includes all the groups and individual "personas" that the device might belong to, and mechanisms that allow managing, adding, deleting, modifying their definitions.

### 7.3.5 Social Tools

The social tools layer deals with peer devices and peer entities – information like trust levels, what to present to them, and any information that a host might keep about its social network or about its perception of its peers' social networks.

The *"Social Register"* is the database that holds the information and settings regarding the peers, groups, and relationships. The *"SocialLeraner"* is the place where learning algorithms could be plugged in for experimentation and integration with the system. The VPM manager is the part that is responsible for managing the shared settings and the information that is emanated from the device. I/O Tools

This layer hosts modules that are related to connecting to input/output of the user as well as remote I/O interfaces. The *GUI Message Interface* module allows user interfaces to connect to the core. The connection between the GUI and the core is done by a socket interface. This allows the GUI to run either on the same machine, or on a remote device that connects through the network.

This is useful for controlling a device remotely, especially in scenarios when the device does not have a screen and/or keyboard. For example, a ubiquitous computing scenario of where the user can walk up to a public computer embedded in the wall, and connect to the screen-less mobile "amulet" device in his pocket. Another example is a printer that supports the Comm.unity framework which the user wishes to configure, for example set its permissions according to the user's own social relationships. Finally, the remote GUI option could be useful even for devices that have their own screens, but they are too small for comfortable

use, like most mobile phones. In this case the user might use a desktop computer to connect to their phone and configure it or browse its Comm.unity application content. The different options to connect the Flash GUI to the Python core are depicted in Figure 7.4, which shows some of the actual implementation blocks. Other GUI platforms other than flash may be connected in the same way through the same socket interface. The interface itself has different types of messages, defined in a "Set"/"Get" format. They allow both the python side and the GUI side to push messages or query for information. In addition, the protocol supports both individual requests ("get me the full metadata on peer ID 'x'"), or bulk request ("get me all data on all peers" currently active"), to allow flexibility and bandwidth efficiency on this link.

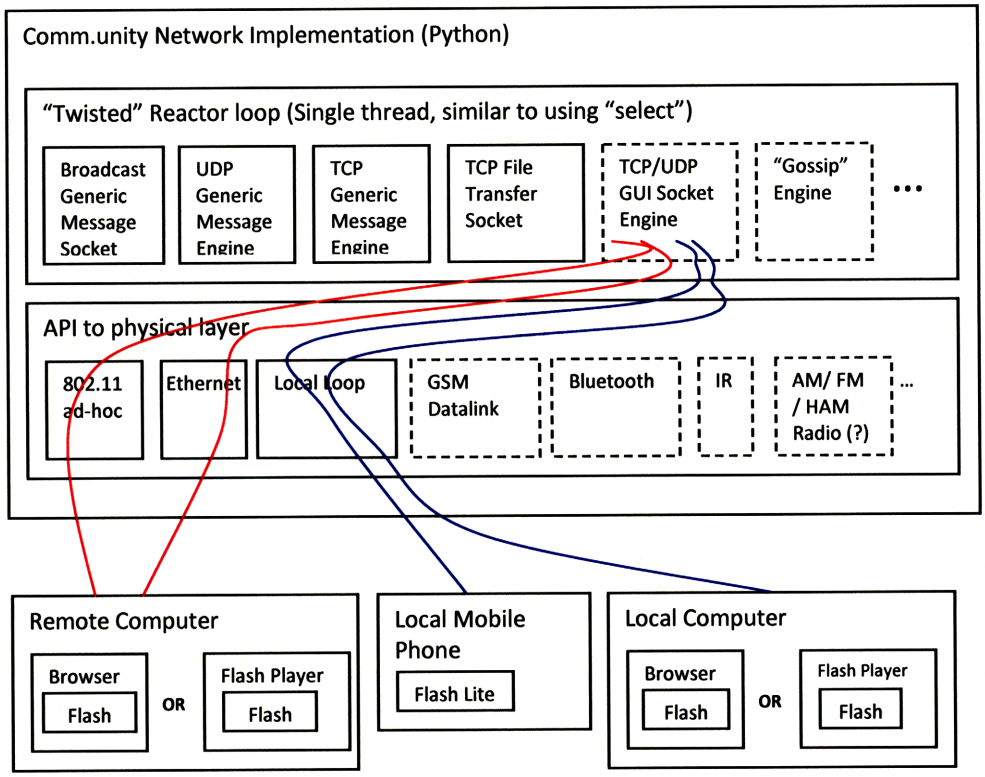



**Figure 7.4 - Comm.unity Implementation: Different ways to connect the flash user interface.**

The *external I/O Interface* implements a more general socket based interface that allows the integration of other types of graphical interfaces or even a general pipe to import and export data and configuration parameters in runtime. For example, connecting Comm.unity to external modules that extract peer proximity information, as well as inject new configurations, like creating new groups and inviting peers in an automated way (without graphical interface intervention).The layer also includes a legacy GTK based GUI module that is integrated into the main loop of the program, which might be useful for platforms that do not support Adobe Flash, although it currently supports older functionality and not the most up to date.

### 7.3.1 Peripherals

The peripherals layer hosts modules that connect to different device peripherals, like camera, audio, or sensors like accelerometers or GPS. One of the ideas behind the API of the layer is that upon startup we could detect the presence or absence of different devices supported by the applications being executed, and make use of them if they are present but not break the execution if they are absent. For example, for logging the user state, if the device has a GPS, we could log that as part of the state logging event, but if it doesn't then this would not break the rest of the scan. Currently most of the peripheral interaction is done through Python, although some of it might be ported to flash for cases where it has easy built-in support, like interacting with a camera or accessing a mobile phone's accelerometer in Flash Light.

### 7.3.2 Task Manager

The task manager is responsible to coordinating and scheduling the tasks and events of the Comm.unity process. For example, deciding when to advertise, when to perform sensor scans, or when save logs to a file, an activity that might be processor intensive and there is a need to make sure that the user experience is not hindered. Another aspect that needs to be manages is when do perform on-device learning, since this too might be a resource expensive activity, especially for a mobile device. This sort of thing might be done, for example, during device charging or when the user is away. The tasks manager will also be responsible for scheduling actions with the high level goal of conserving battery life.

### 7.3.3 Application manager

The application manager is the point where external applications resister themselves with the Comm.unity core. The appropriate applications would then get notified by relevant events, like new content files added to the system, information about peers discovered, peer messages, and so on. Respectively, applications are able to send information to Comm.unity, for example a camera application can notify the core that a new picture was taken and then initiate a process to add it to the Comm.unity content collection (e.g. generate the relevant hashes and metadata, and associate it with the appropriate entities).

### 7.3.4 Data Collection Engine

This layer is responsible for logging aspects and data collection from all parts of Comm.unity. this data will later be aggregated and parsed, either by the device itself or propagated to a central server (for research purposes). The data is saved in JSON format ["Introducing JSON,"] to make it easy to save to a file, parse, modify, and transfer over the network.

### 7.3.5 Context/State Manager

The context and state manager hosts the algorithms that decide on the current context and "state of the world", which in turn affect the device's behavior. For example, determining which VPM entities to activate at any given moment. Context could be as simple as "home" vs. "work", or much more complex constructs.

### 7.3.6 Flash Front-End

Figure 7.5 shows the main component of the Adobe Flash side of the Comm.unity implementation. *Comm.unity Central* is the flash module that acts as the "main" program, and is responsible for loading and initializing all Flash related modules and applications. Developers who develop applications that are solely Flash-based or have some Flash component would register their application with this module. The *Network Engine* flash module aggregates the communication with the Python code, and communicates directly with the I/O layer's *GUI Message Interface*. The stateClass keeps track of the current state of the system and the different applications. It caches information that arrives from the Python side and allows different applications to access it, and also keeps track of GUI related information that is not application specific (for example configurations of the main window size, etc.). Finally, the *Status Console* acts as a Flash based terminal window that prints out any messages that detail the status of the current execution as well as for debugging purposes.

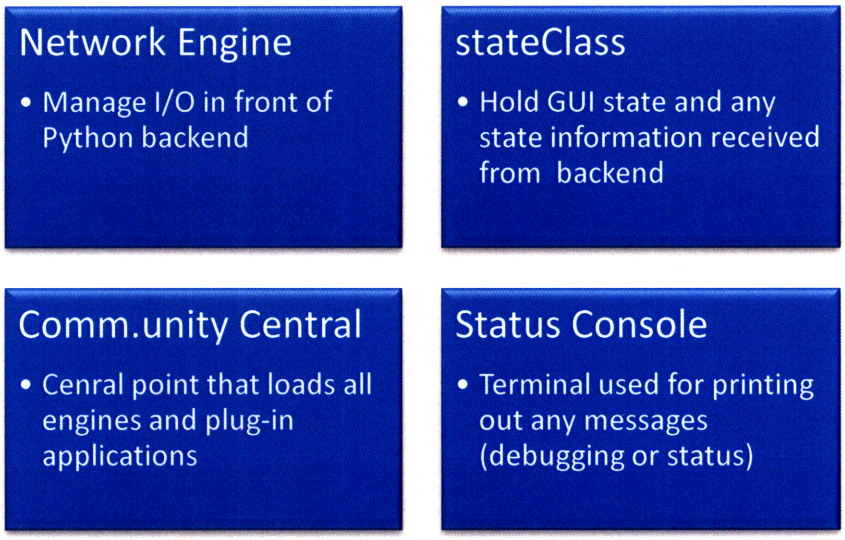



Figure 7.5 - Flash front-end main components.

# 8 Implemented Social Applications

*Few things are harder to put up with than the annoyance of a good example.*

*-Mark Twain, The tragedy of Pudd'nhead Wilson*

In this chapter we review the two main user applications that have hitherto been developed over Comm.unity. The first is the ***Social Dashboard application***, which implements the dashboard interface approach described in chapter 5. The dashboard acts as a bridge between the end-user and the many concepts described earlier in the thesis. The second application is **SnapN'Share**, which enables users to generate and seamlessly share content with different groups and communities that they belong to, as they come within close vicinity of their peers. Both applications serve as a podium for demonstrating the ideas introduced in this work and bringing them to the hands of end-users, as well as a 'playground' and laboratory for further development of these ideas and new ones.

## *8.1 The Social Dashboard Application*

The Social Dashboard (SD) user application is intended to serve as a platform for implementing, exploring, and evaluating the myriad of concepts that were presented throughout the previous chapters of the thesis – including social interaction in the hybrid space, allowing users to control their digital presence and content, configuring the behavior of users' devices and applications through a social metaphor, autonomous (or semi-autonomous) learning of social structures such as social networks, groups, relationships and levels of trust. The SD integrates with the other building blocks that we have designed and implemented, specifically the VPM framework, for which it acts as the main user interface, and the Comm.unity platform, over which it is implemented.


Figure 8.1 - Social Dashboard Logo

### 8.1.1 Implementation Overview

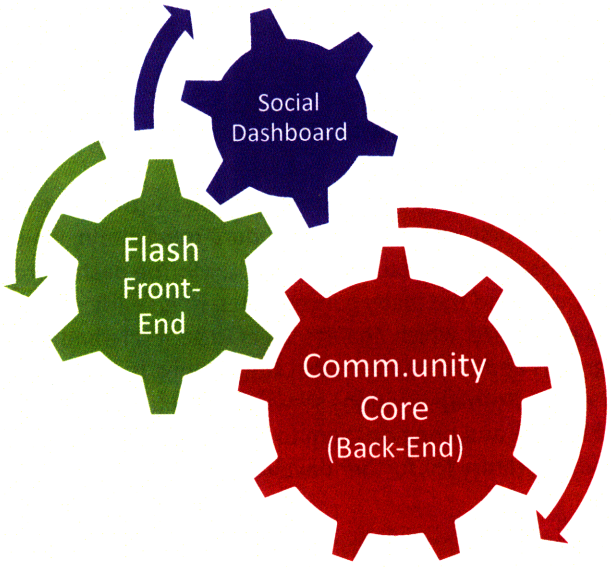



Figure 8.2 - Social Dashboard integration with Comm.unity through Flash front-end API

The social dashboard integrates with the Comm.unity platform at the Flash interface. It builds on the Social Dashboard interface concepts described in chapter 6, but also implements other functionality to make it a usable user application. The Social Dashboard application can function both as a standalone program or as a module integrated with user defined programs. For example, a music sharing application[16] might add an action to a peer entity's menu which would allow the user to brows a peer's music collection as they sit together on the subway. Other functionality that the dashboard offers include an ability to chat with peers, create and manage groups that the user belongs to, invite peers to a group by dragging them unto it, ability to move peers around and notify the Comm.unity core of their new trust setting, a chat interface, and a manual file sharing interface: The user can select or "pre-load" a file to a designated icon (can be seen at the bottom left corner of Figure 8.3), and then drag the icon onto an active entity in its vicinity in order to initiate a file transfer. The receiving side can choose whether to accept files automatically, ask the receiving user for confirmation, or reject automatically – according to the sender's placement on the dashboard. It is implemented in Adobe Flash and is currently written in ActionScript 2, but we are currently developing an Actionscript 3 upgrade. The application relies on the Comm.unity core to store a memory of its configuration settings, as well as memory of peers that the device has already come across in the past.

Current implementation has been tested and demonstrated in a live event using 16 devices in wireless ad-hoc mode. The devices included a window XP laptop, 12 Nokia N810s, and 3 Nokia N800 devices. Figure 8.4 shows the dashboard in action in a 12 device experiment. The laptop screen shows the 11 peer devices and 4 group entities that the laptop is part of.

[16] Intended for only for LEGITIMATE use.

Potential usage scenarios of the current version include, for example: peers discovering presence of each other in a distributed way as they roam the world; interactions for conference settings, where users exchange personal information, contact information, and data; peers interacting with each other in a lecture or class setting, both with each other and with the lecturer. The lecturer may also use the system to hand out digital material to the crowd.

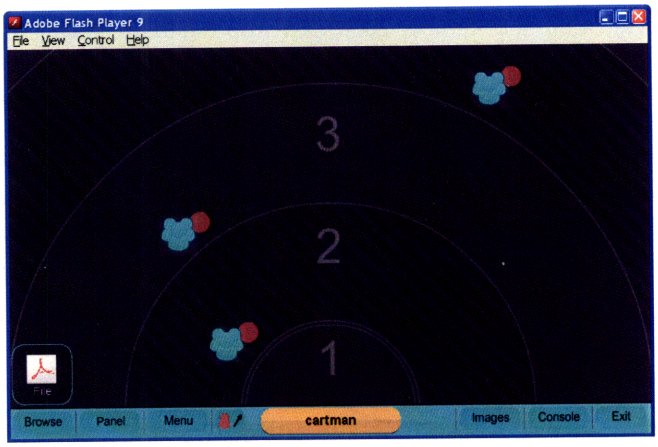



Figure 8.3 - Social Dashboard Screenshot

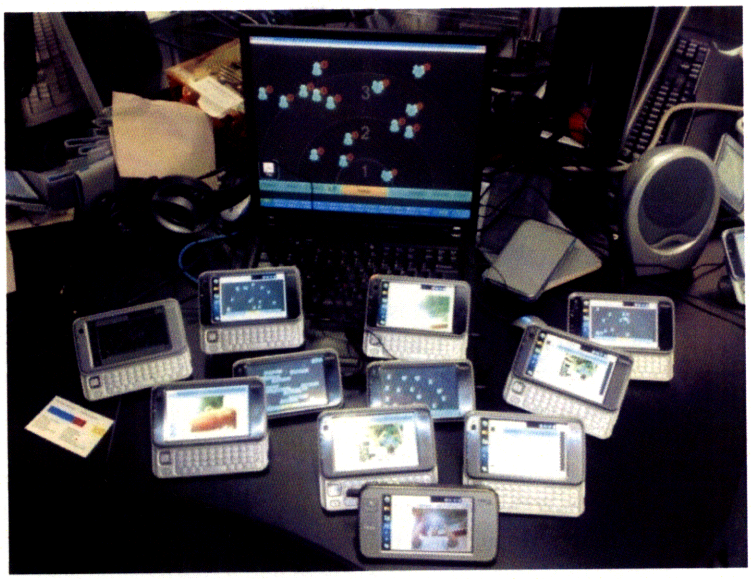


Figure 8.4 - Social Dashboard and SnapN'Share test with 12 active devices in ad-hoc mode.

## 8.2 SnapN'Share and Proximity Blogging: Sharing Our Digital Aura off the Grid

The SnapN'Share application [Aharony et al., 2008], built on top of the Comm.unity platform, enables users to generate and seamlessly share content with different groups and communities that they belong to, as they come within close vicinity of their peers. The

owners, creators, and viewers of the information are also the ones holding it on their personal devices. SnapN'Share was the first application developed with Comm.unity, and in fact it motivated the creation of the Comm.unity platform and initiated the thinking process for many of the ideas presented in this thesis. The system is used to demonstrate many of Comm.unity's features and aid in the platform's development. It is planned to be released as a downloadable user application in the near future.

SnapN'Share started around December 2006, as a proposal to the Knight Foundation's 21st Century Challenge [Knight], suggesting to create a peer-wise content dissemination platform that lets news, publications, and user content migrate through a community as people physically pass by each other. It was submitted as an individual grant request and also as part of MIT's proposed Center for Future Civic Media (C4FCM) initiative. The individual proposal came out as the runner up of the $5M challenge. The C4FCM proposal was the grand winner of the challenge, and the system is now part of the center's portfolio.

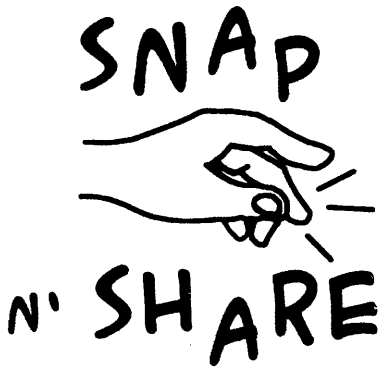



**Figure 8.5 - SnapN'Share Application Logo**

### 8.2.1 Overview

The SnapN'Share application runs on wireless mobile devices as well as other devices, like laptops or PCs. It allows users to generate and seamlessly share content with different groups and communities that they belong to and have face-to-face interactions with, such as friends, family, co-workers, or neighbors. The groups can be easily created and sustained for long or short periods. "Family" could be an example for a more permanent group, whereas an example of a shortly lived group could be a number of people taking a trip together and desiring to easily and immediately share digital photographs that different people are taking along the trip. Another example could be a team assembled for a specific assignment, like a class project or a journalistic story, and want a way to easily share digital content with their peers. A user would simply walk into a meeting room and the meeting's presentation would seamlessly "hop" onto their laptop or PDA. Attendees could take notes or record sound snippets and share them with their peers. They could even add content after the meeting, such as documents and URLs that are relevant to that context, and those will also be seamlessly shared with the others at the next time they are physically nearby. Finally, users may choose to make some content available to anyone around them – e.g. the general public. This could be used to listen to music off of peers' devices on the subway, publicize a personal website or blog, or distribute other information that is relevant in a certain locale - advertisements, notifications, or a localized, ad-hoc bulletin board. Stationary nodes could

also be part of the system, offering information and services to users as they enter their vicinity, thus supporting simple implementations of location and context aware applications.

### 8.2.2 Vision

Imagine a platform that lets news, publications, and user content migrate through a community in a flurry of direct, person-to-person interactions. SnapN'Share is intended to introduce a form of content publication and dissemination where the information is physically held in common by all. It can be thought of as a "hot potato" with a trail - information is shared among people as they quite literally pass by each other, leaving pieces of information in personal, portable devices. The owners, creators, and viewers of the information are also the ones holding it on their personal devices, rather than requiring external service providers for this purpose.

### 8.2.3 Implementation

Any device that runs the application is part of the SnapN'Share wireless ad-hoc network. This ad-hoc network acts as a VPM's top-setting, a rendezvous point for any user device that runs the system. As devices come into range of one another, they discover each other and perform the VPM discovery process and handshake session to determine if they have any content groups in common. If they do, they would exchange the list of files that each of them holds for that group. Each will then have the option to request any group files and content that they currently do not posses. For each group they belong to, or for each peer that they come in contact with, users have the ability to configure how automated the content exchange would be. For some peers, with whom there is a highly trusted relationship, the user would accept content automatically. For other peers or groups the user would choose to be prompted before receiving content. In the presence of certain peers, users would be required to specifically request information, avoiding the possibility of unwanted content from un-trusted groups or peers.

Users have the ability to dynamically create new groups and associations, as well as add content to the groups. The system's user interface displays to the user, in a graphical way, the different users and groups that are present within a wireless 'hop' away (using the Social Dashboard module described above). The left side of Figure 8.6 shows SnapN'Share in action, in a test run involving 14 devices. The left side of Figure 8.6 shows a close-up on the picture "snapping" interface. It allows the user to take pictures, and also designated which groups or he would like associate the next photograph with, or any subsequent photos. Once taken, the photograph is linked to the respective virtual spaces of the designated groups. Pictures are just one example for demonstration purposes, and any other type of content capture or import may be added to the system. Figure 8.7 depicts the process of how content spreads throughout the members of a group in a hop-by-hop manner. It would also spread to any other member of the relevant groups who is not currently present. Whenever a member of a group passes by any other member, their respective virtual spaces would synchronize with any missing content. These synchronized spaces create a content network which is off-the-grid with respect to regular infrastructure. We dub this *"proximity blogging"*. Users can view and interact with their peers on their mobile devices. Alternatively, they could use Comm.unity's remote interface option to connect to their mobile devices through a PC with a larger screen and a full sized keyboard. Another option is to use the VPM framework to define a fully trusted group between their personal devices, and offload any

content from their mobile device to their home computer as they come within range of it. They could then interact with the content directly on their home machines. In addition, it could be easily configured that a device that also has access to the regular infrastructure would post the content to a server or a designated website, to allow remote users to access the material or for archiving purposes.

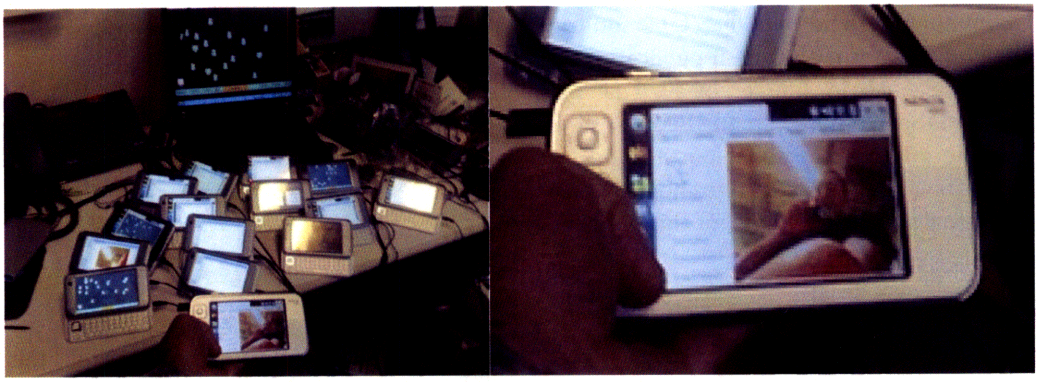


Figure 8.6 - Left: SnapN'Share test run with 14 devices; Right: Picture taking interface.

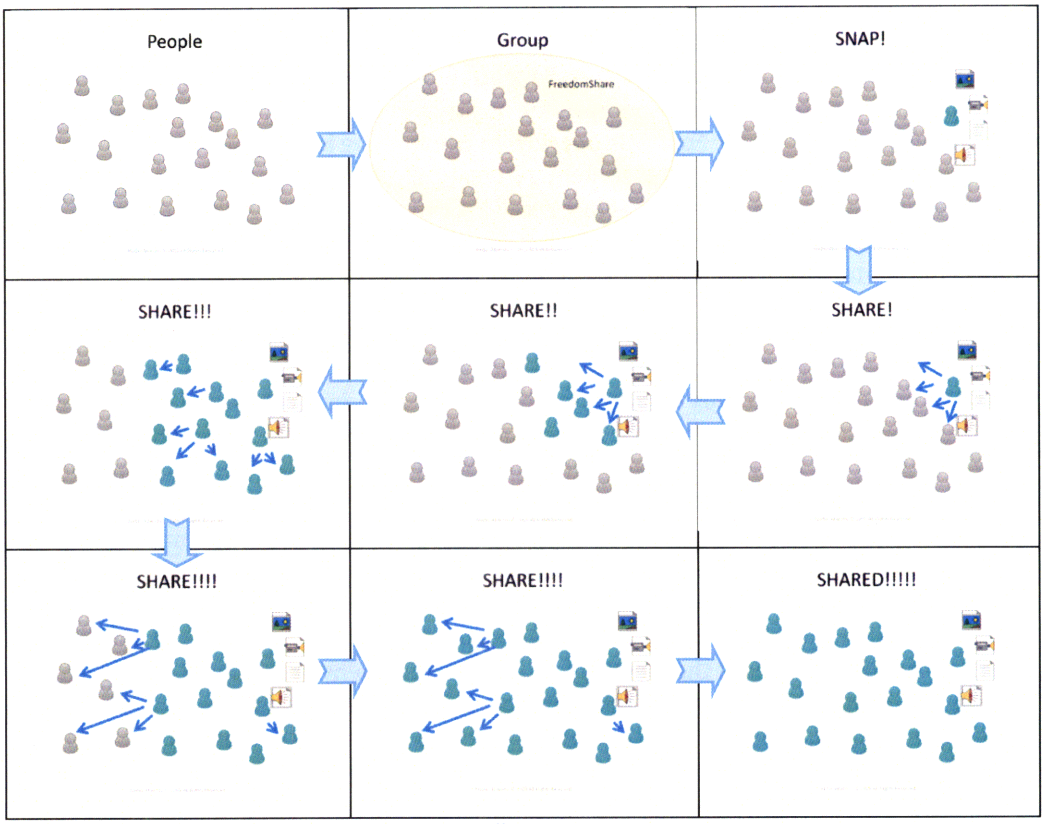



Figure 8.7 - The 'Share': Content dissemination illustration.

Figure 8.8 shows the current content viewing, or *proximity blog* interface. Inspired by Castell's Space of Flows, each group's virtual space is composed of multiple flows. Each flow

can represent a topic, an event, or any other subject that group content is generated around. Once that topic becomes stale or irrelevant, it may be archived and not synchronized. Clicking on a group in the group list will expose its flows. Clicking on a flow will show the items that are part of that flow. Clicking on each item will show the item, including a thread of comments that are linked to the item. As shown at the bottom left of Figure 8.8, users can post comments to any item, and those comments will be treated as new content that needs to be disseminated through the group. Each space, flow, or item has associated metadata, and there are various ways to sort the list of objects in each view. For example – highlighting the newer and more active objects and sorting the list accordingly. Aside from just the content, an interesting direction to explore is sharing other metadata among peers, for example which items are viewed more than others, and use this information to create a more "social" sorting of the list according to the group member's interests. There are multiple ways to implement a deletion of an item. One way would be to allow users to physically delete an item and never request it again from peers. Another way would be to implement the deletion by marking that the item is deleted and removing it from the user interface, but it would still be stored on the user's device and disseminated further.

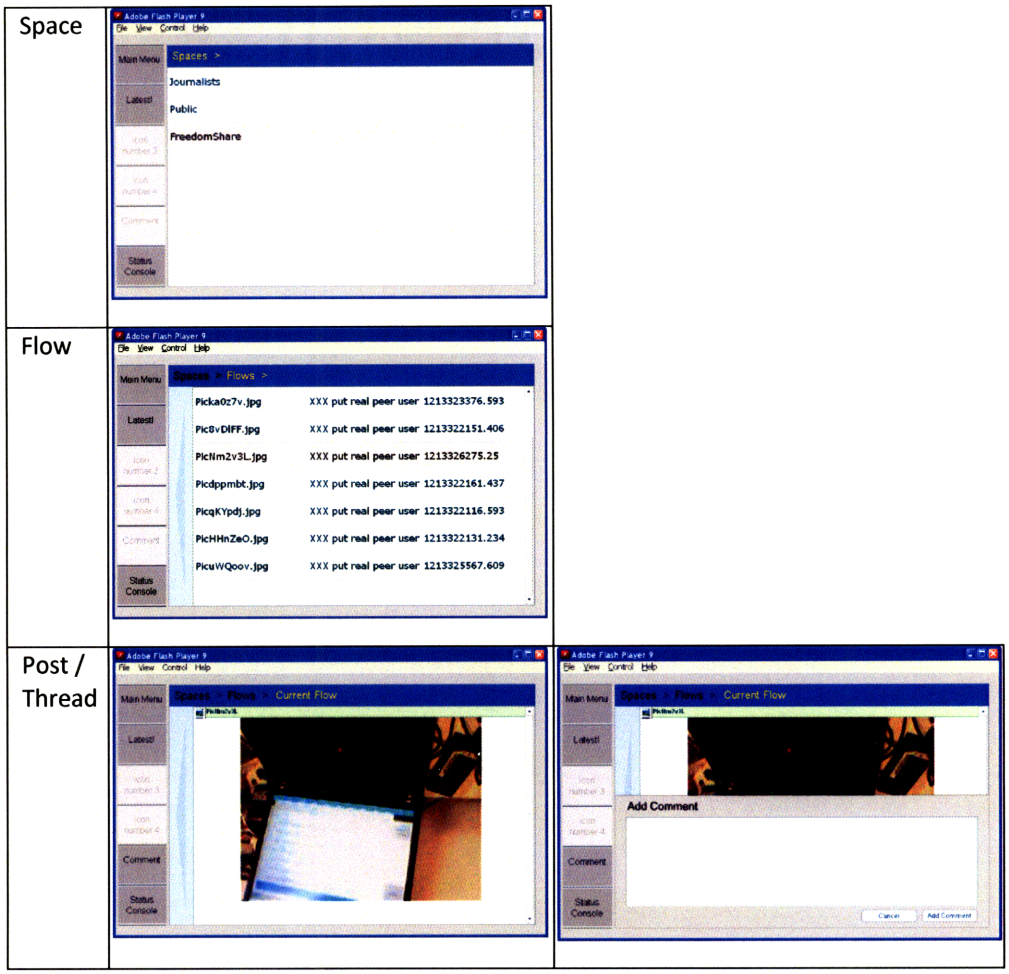



**Figure 8.8 - Content viewing interface ("Proximity Blog"). Top: Flow display; Middle: Thread display. Bottom left: Single post interface. Bottom right: Adding comment to post.**

# 9 Conclusion

*"If you want to build a ship don't herd people together to collect wood and don't assign them tasks and work, but rather teach them to long for the endless immensity of the sea."*

*-Antoine de Saint-Exupery, the Wisdom of the Sands*

## *Onward*

One possible direction that this work could lead to in the area of networking technologies relates to the ideas described in The *Society of Network* [Aharony, 2007]. In that document, I describe a vision of autonomous network devices, from edge devices to devices at the heart of the network infrastructure. These devices combine artificial intelligence inspired by Minsky's Society of Mind [Minsky, 1988] and Emotion Machine [Minsky, 2006] combined with the metaphor of human social interaction. In addition to the relationships of a human user, these devices create relationships and communication of their own, with peer devices. The devices interact with their peers and their environment as people do, creating relationships of trust or distrust, notions such as "family" or "friendship. At the time I realized that we need a lot more knowledge about human society before we try to emulate it.

These thoughts, combined with the SnapN'Share proposal described in the previous chapter, evolved into what has become the work in this thesis. *Hopefully, we could reach a stage where we can implement all of our acquired knowledge of human society to create a new breed of autonomous network devices that act as a society, are inspired by human society, and are as flexible and robust as human society.*

The ideas of the Society of Network would only relevant in a distant future, and only if we choose to pursue them. There are many more useful scenarios that we could enable in the immediate future and are directly relevant to human users in everyday life:

- Imagine going on a trip and creating an ad-hoc group with the people you happen to be traveling with. Any picture anyone takes could be immediately distributed to all of the group's devices. Imagine being able to chat with strangers on the plane, or a friend sitting ten rows in front of you in a lecture hall.
- Imagine taking the subway and getting the digital version of the Metro paper as you walk by the T-stop. Then, share music and files with strangers sitting next to you in the subway car.
- If you are worried about getting spam and viruses from strangers, you could use information about who you know and trust to filter out unwanted peers.
- Imagine having "virtual spaces" that you can easily create and share content with different groups in your life – friends, family, classmates, co-workers, or whatever you choose.
- Finally, how about being able to get notifications when your colleagues, friends, and family are nearby?

Let us back up, and think more globally: How about a communication system for emergencies and disaster scenarios, which allows news and critical information to be distributed among people in the area without the need for existing infrastructure (which is likely to be down...)? Or perhaps a system for professional or civic journalists that are operating under oppressive regimes? These are just a few scenarios out of countless others.

Now imagine doing all of the above for free, no service charges, over an open platform that would allow any developer to enhance and add new features and applications.

## *Overture*

In this thesis we presented architectures and tools that aim to improve our connection to other people and devices around us, as well as improve our control and understanding of our own communication devices.

We started by carving out a limited domain out of the networking space – close proximity networking. We looked at both the human aspects and technological aspects of close proximity and face-to-face interactions. From the human side, we tried to highlight the characteristics, needs and potential caveats of FtF interaction in today's highly mobile and networked society. From the networking perspective we attempted to justify the need to aggregate the handling of close proximity communication technologies under one abstraction, which we dubbed a *Face-to-Face Network*, or *FtFN*. We used the social discussion to help us frame the requirements and services that an FtFN abstraction layer should support.

Next we analyzed the interaction aspects of social applications for close-proximity digital technologies. We argue that this interaction space should be treated in a different way than the traditional treatment for either human interaction or digital interaction. This interaction space is unique in the sense that the network device and the users themselves act as a physical anchor to their projected digital aura. The user's physical signals and cues can affect the digital interaction, and vice versa. This is why we dubbed this for our discussion as *the hybrid space* , which is the unison of Castell's space of places and space of flows [Castells, 1996]. We discussed different characteristics of this space, as well as some of the risks or social awkwardness that could occur due to hasty design of systems and applications. Finally we gave our proposal for preliminary interaction design principles for the hybrid space.

We then moved on to implement those designs in our proposed *Virtual Private Milieu* (VPM) framework and the *Social Dashboard* interface concept. The goal of the VPM framework is to provide a robust solution for an unmanaged, group-based discovery and communication architecture among entities with access to a shared communications medium. Key features of this framework include the incorporation of trust and context parameters into the discovery and communication process, support for multiple, context-unique identities, as well as support of multiple degrees of security and privacy settings.

The *Social Dashboard* presents a novel concept of human-network interface, as well as device management interface. It is our proposed solution for an interface aimed at FtFN . It allows users to discover peers and services in their social and physical vicinity based on the VPM architecture, and communicate with them using a provided set of interaction tools aimed to support face-to-face digital interactions. In addition, the dashboard is also an interface that allows users to configure and control the behavior of their device towards

these peer devices and services, based on their social connection to the owners of the devices, and based on social metaphors in general. The Social Dashboard is based on ideas from the area human centric design and cognitive load theory. Rather than attempt to lay out peer entities as they are laid out in the physical world, we try to represent them in a radar-like interface according to how they might be laid out in the user's mind – along axes of social relationships and trust.

Next we broadened our discussion to applying the ideas of socially inspired network behavior to networking and digital communications in general. We proposed *"Social Area Networks"* (SocANs) - a concept encompassing network architectures built for and around people and their social relationships. In these architectures, the users' social information pervades all the way through the network stack, down to the lower layers of the network, and is used for configuring the network's behavior and its services to the users. In a complementary manner, network logs and status data could be used for inferring social information.

At this point we turned to show that these ideas are not just feasible, but many parts of them are already implemented. We gave an overview of the *Comm.unity* platform, the software package developed through this project that encases the VPM framework and enables easy development and deployment of a variety of socially informed applications for sharing information and content among peers who are in close vicinity of one another. It is intended to be used by both application developers and researchers alike. Finally, we described the two main user applications that have hitherto been developed over Comm.unity. The first is the *Social Dashboard application*, which implements the dashboard interface approach described earlier. The second application is SnapN'Share, which enables users to generate and seamlessly share content with different groups and communities that they belong to, as they come within close vicinity of their peers. Both applications serve as a podium for demonstrating the ideas introduced in this work and bringing them to the hands of end-users, as well as a 'playground' and laboratory for further development of these ideas and new ones.

*It is my hope that this work would help improve the connection between network media and human media, and inspire those who delve in either of those realms to seek out knowledge and better understanding of the other.*

# 10 Appendix: Understanding Media and Ourselves

In this section we briefly mention some seminal works relevant to our interest in new media and its effect on society. These are relatively high-level theories, on the essence of media itself, and about life in our modern, media-rich and networked society.

## *Network Media as Extensions of Ourselves*

In *Understanding Media*, Marshall Mcluhan theorizes that all media are extensions of Man's physical and psychological being that seek to increase speed and power. This is a dual relationship, as media, in turn, shape Man's physical and psychological being. [McLuhan, 1964]

According to McLuhan, the "gadget", or electronic technology in general, is an extension of man. Its use can lead to both infatuation and narcosis, or numbness. Any technology is a sensual experience, and as such it affects the physical and emotional awareness of people. Technologies cause "Man" to become more fragmented, and more susceptible to technologically induced physical as well as emotional change. We embrace technological extensions of ourselves simply by perceiving them (p.46). At the same time that media is an extension of man, it could also amputate man, as we mentioned earlier. McLuhan's notions all lead to the important need of people - users, designers, and developers alike, to be informed and aware of the powerful effects that media can cause.

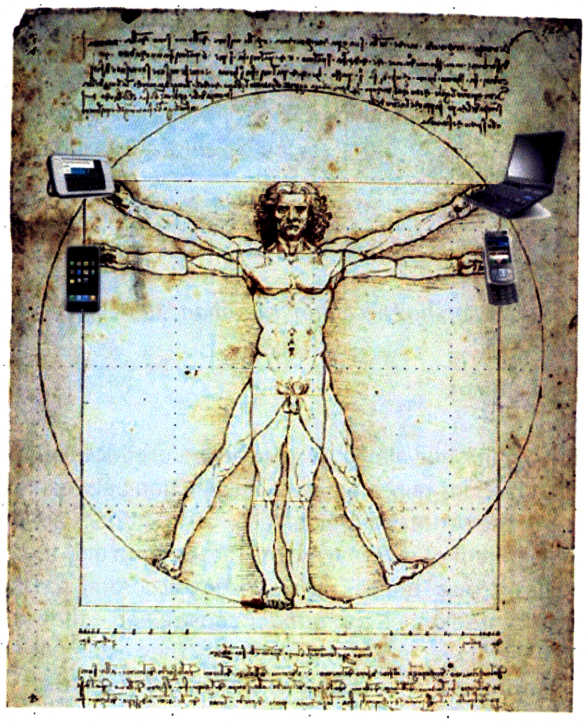


Figure 10.1 - Mobile network devices as extensions of man.

## *Ergo, the NETWORK is the Message. Q.E.D.*

We already quoted Marshall McLuhan's saying of "The medium is the message" in the introduction section. The premise of this thesis is that our communication networks themselves are media. In fact, they are very generic "wildcard" media, since just like light, they can be "pure information" [McLuhan, 1964, p. 8]. In his book Mcluhan splits and divides the different types of new media, like TV, currency, radio, or the telephone. Data networks are a platform that enables all of these different media. Sent over the network, all of these media become content themselves (as the message of the network), that carries further encapsulated media and content. This fits well with McLuhan's discussion of the nesting of media within media: "The content of writing is speech, just as the written word is the content of print, and print is the content of the telegraph. If it is asked, 'What is the content of speech?' it is necessary to say, 'It is an actual process of thought, which is in itself nonverbal.' " [McLuhan, 1964, p. 8].

Aside from their conventional encapsulated content being transmitted, the network provides signals of physical and social presence (e.g. by emanating radio waves in certain locations at certain times), social relationships (e.g. by source and destination of packets), and many other signals related to the users and the state of the world. Application's like Facebook "mini-feed" [Facebook] generate content from logging seemingly trivial user actions in the online realm (adding friends, viewing content, playing some game, etc.) and turn it into a narrative.

Today, more than 50 years after McLuhan's seminal work was published, there remains an implicit and many times *explicit* separation between network "content" and the medium of the network itself, the way we think about it and design it. Many times there is also a strict distinction between the type of people who develop for the network's infrastructure and the people who create content and application to be sent through these networks. Perhaps there is justification to educate our "pure engineers" with such kinds of media related work.

> *"Indeed, it is only too typical that the 'content' of any medium blinds us to the character of the medium."*
>
> *-Marshall Mcluhan [McLuhan, 1964, p. 9]*

## *People are Media Too*

In "Thing and Medium", published almost forty years before McLuhan described light as media, Psyhcologist Fritz Heider talked about the distinction between two types of entities - "things" and "media". In the context of Heider's discussion, things are *internally* constrained, and are relatively independent of external events for the form and the energy distribution within them. A stone has strong "thing" characteristics, according to Heider. Conversely, "media" present a high degree of being *externally* constrained – they are relatively dependent on external events for the form and energy they exhibit. He gives the example of light rays reflected from stone as an example of a manifold of entities with high medium characteristics – the light pattern is determined in some way by the external stone [Heider, 1926].

Barker uses Heider's terminology in the context of *"behavior settings"*, which we discuss in detail in chapter 4. For now, we can think of a behavior setting as a confined physical interaction space that has some purpose, like a store or a baseball game, and affects the behavior of its inhabitants. With relation to our current context, Barker says that "People are the media of behavior settings." [Barker, 1978, p. 217].

People are essential to behavior settings setting, and without it they do not exist. In the same way, we can think of scenarios where people are actually the media of the networks – the most trivial example of which is the idea of network routing by social network [For example, see Ford et al., 2006; Marti et al., 2004], where the network uses people as 'mules' and carriers of data as they move about the physical world and interact with their social peers.

## *Go With the Flow*

In "The Rise of the Network Society", Manuel Castells defines the *space of* flows [Castells, 1996]. In his words, he points out that "... our societies are constructed around flows: flows of capital, flows of information, flows of technology, flows of organizational interactions, flows of images, sounds and symbols. Flows are not just one element of social organization: they are the expression of the processes dominating our economic, political, and symbolic life. ... The space of flows is the material organization of time-sharing social practices that work through flows. By flows I understand purposeful, repetitive, programmable sequences of exchange and interaction between physically disjointed positions held by social actors. " (p.412).

As Castells explains, the space of flows allows us to transcend the limits of time and space, in contrast to the *space of place*, which is bounded by them. Castells talks about how the space of flows connects between distant locations thus reconfiguring the space of place. The interaction and mutual reconfiguration of these two spaces is a main theme of this thesis as we discussed in chapter 3 – the hybrid space that emerges from the union of Castells' spaces.

# References


Abadi, M., & Fournet, C. (2004). Private authentication. *Theor. Comput. Sci., 322*(3), 427-476.

Adjie-Winoto, W., Schwartz, E., Balakrishnan, H., & Lilley, J. (1999). *The design and implementation of an intentional naming system*. Paper presented at the Proceedings of the seventeenth ACM symposium on Operating systems principles.

Adobe. Flash Player Penetration. from http://www.adobe.com/products/player_census/flashplayer/

Aharony, N. (2007). The Society of Network. Massachusetts Institute of Technology.

Aharony, N., Lippman, A., & Reed, D. P. (2008). *Demonstrating SnapN'Share: Content Sharing over a Socially Informed, Hyper-Local, Face-to-Face Networking Platform.* Paper presented at the 5th IEEE Consumer Communications and Networking Conference, 2008. CCNC 2008. , Las Vegas, NV.

Andersen, D. G., Balakrishnan, H., Kaashoek, F., & Morris, R. (2001). *The Case for Resilient Overlay Networks*. Paper presented at the Proceedings of the Eighth Workshop on Hot Topics in Operating Systems.

Anderson, P. W. (1972). More is Different. *Science* (177), 393--396.

Aspan, M. (2008, February 11). How Sticky Is Membership on Facebook? Just Try Breaking Free. *New York Times*, from http://www.nytimes.com/2008/02/11/technology/11facebook.html?ref=todayspaper

Barker, R. G. (1978). *Habitats, Environments, and Human Behavior: Studies in Ecological Psychology and Eco-Behavioral Science from the Midwest Pssychological Field Station, 1947-1972*. San Francisco, CA: Jossey-Bass, Inc. Publishers.

Barry Wellman. (1992). "Men in Networks: Private Community, Domestic Friendships. In P. Nardi (Ed.), *Men's Friendships* (pp. 74-114). Newbury Park, CA: Sage.

Barth, A., Boneh, D., & Waters, B. (2006). *Private encrypted content distribution using private broadcast encryption.* Paper presented at the 10th Financial Cryptography and Data Security Conference (FC 2006).

Blanchard, A. (2004). Virtual Behavior Settings: An application of behavior setting theories to virtual communities *Journal of computer-mediated communication 9*(2).

Bloom, B. H. (1970). Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM, 13*(7), 422-426.

Bluetooth SIG. Bluetooth Specification v1.1, Part E: Service Discovery Protocol (SDP).

Bluetooth SIG. How Bluetooth Technology Works. Retrieved July 22, 2008, from http://www.bluetooth.com/Bluetooth/Technology/Works/

Bluetooth SIG. (2007). Bluetooth Specifications (V2.1). from http://www.bluetooth.com/Bluetooth/Technology/Building/Specifications/

Boase, J., Horrigan, J. B., Wellman, B., & Rainie, L. (2006). *The Strength of Internet Ties*. Washington, DC.: Pew Internet & American Life Project.

Borovoy, R., Martin, F., Vemuri, S., Resnick, M., Silverman, B., & Hancock, C. (1998). *Meme tags and community mirrors: moving from conferences to collaboration.* Paper presented at the Proceedings of the 1998 ACM conference on Computer Supported Cooperative Work.

Boyan, J. A., & Littman, M. L. (1993). Packet routing in dynamically changing networks: A reinforcement learning approach. In J. D. Cowan, G. Tesauro & J. Alspector (Eds.), *Advances in Neural Information Processing Systems* (Vol. 6, pp. 671-678). San Francisco CA: Morgan Kaufmann.

Boyen, X. (2007). *Mesh Signatures.* Paper presented at the Advances in Cryptology - EUROCRYPT 2007.

Brown, T. X. (2001). *Switch Packet Arbitration via Queue-Learning.* Paper presented at the Neural Information Processing Systems, Vancouver, BC.

Calhoun, C. (1998). Community without Propinquity Revisited: Communication Technology and the Transformation of the Urban Public Space. *Sociology Inquiry, 68*(3), 373-397.

Campo, C., Garcia-Rubio, C., Lopez, A. M., & Almenarez, F. (2006). PDP: a lightweight discovery protocol for local-scope interactions in wireless ad hoc networks. *Computer Networks, 50*(17), 3264-3283.

Caron, A. H., & Caronia, L. (2007). *Moving Culture: Mobile Communication in Everyday Life.* Montreal: McGill-Queen's University Press.

Castano, J. G., Svensson, M., & Ekstrom, M. (2004). *Local positioning for wireless sensors based on Bluetooth.* Paper presented at the Radio and Wireless Conference, 2004 IEEE.

Castells, M. (1996). *The Rise of the Network Society*: Blackwell Pub.

Cerf, V., Burleigh, S., Hooke, A., Torgerson, L., Durst, R., Scott, K., et al. (2007). *Delay-Tolerant Network Architecture, IETF RFC 4838.*

Chandra, R., Mahajan, R., Padmanabhan, V., & Zhang, M. (2006). CRAWDAD data set microsoft/osdi2006 Retrieved?????, from http://crawdad.cs.dartmouth.edu.

Chaum, D., & Heyst, E. v. (1991). *Group signatures.* Paper presented at the Advances in Cryptology - EUROCRYPT '91.

Cheshire, S. (2005). Zero Configuration networking with Bonjour.

Cheshire, S., & Krochmal, M. (2006). *DNS-Based Service Discovery (Internet-Draft)*: IETF.

Cheshire, S., & Steinberg, D. H. (2005). *Zero Configuration Networking: The Definitive Guide.*: O'Reilly Media.

Clarke, D., Elien, J.-E., Ellison, C., Fredette, M., Morcos, A., & Rivest, R. L. (2001). Certificate chain discovery in SPKI/SDSI. *Journal of Computer Security, 9*(4), 38.

Clarke, D. E. (2001). *SPKI/SDSI HTTP server : certificate chain discovery in SPKI/SDSI.* Massachusetts Institute of Technology.

Cohen, B. (2003). *Incentives build robustness in BitTorrent.* Paper presented at the Proceedings of the First Workshop o the Economics of Peer-to-Peer Systems, Berkeley, CA.

Cox, L. P., Dalton, A., & Marupadi, V. (2007). *SmokeScreen: Flexible Privacy Controls for Presence-Sharing.* Paper presented at the MobiSys 2007, Puerto Rico.

Czerwinski, S. E., Zhao, B. Y., Hodes, T. D., Joseph, A. D., & Katz, R. H. (1999). *An architecture for a secure service discovery service.* Paper presented at the Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking.

dataportability.org. Retrieved August 5, 2008, from http://dataportability.org/

Delver. Retrieved August 5, 2008, from http://www.delver.com/

Dodgeball. Retrieved July, 2008, from www.dodgeball.com

Doi, T. (1973). *The Anatomy of Dependence: Exploring an area of the Japanese psyche - feelings of indulgence*: Kodansha International Ltd.

Donath, J. (Fourtcoming). Signals, cues and meaning. In *Signals, Truth and Design*: MIT Press.

Eagle, N., & Pentland, A. (2005). Social serendipity: mobilizing social software. *IEEE Pervasive Computing*, 28-34.

Eagle, N., & Pentland, A. (2006). Reality mining: sensing complex social systems., *10*(4), 255-268.

Erving Goffman's Obituary. (1982, November 22). *New York Times*,

Facebook. Retrieved August 5, 2008, from http://www.facebook.com
Facebook. Facebook Developer Site. Retrieved August 5, 2008, from http://developers.facebook.com/
Fette, B. A. (2006). *Cognitive Radio Technology*: Newnes.
Fettig, A. (2005). *Twisted Network Programming Essentials*: O'Reilly Media, Inc.
Foley, M. (2007). Wibree Press Release. from http://www.bluetooth.com/NR/rdonlyres/9A00E8AB-1337-40FF-A4A5-4092D9E17121/0/BluetoothLowEnergyTechnology_MikeFoley_TheNewWirelessFrontier.pdf
Ford, B., Strauss, J., Lesniewski-Laas, C., Rhea, S., Kaashoek, F., & Morris, R. (2006). *Persistent Personal Names for Globally Connected Mobile Devices.* Paper presented at the 7th USENIX Symposium on Operating Systems Design and Implementation (OSDI '06), Seattle, Washington.
Foth, M., & Hearn, G. (2007). Networked Individualism of Urban Residents: Discovering the communicative ecology in inner-city apartment buildings. *Information, Communication & Society, 10*(5), 749 - 772
Fragouli, C., Katabi, D., Markopoulou, A., Médard, M., & Rahul, H. (2007, October). *Wireless Network Coding: Opportunities & Challenges.* Paper presented at the 2007 Military Communications Conference (MILCOM 2007), Orlando, FL.
Freedesktop.org. D-Bus. Retrieved August 5, 2008, from http://www.freedesktop.org/wiki/Software/dbus
Freedman, M., & Nicolosi, A. (2007). *Efficient private techniques for verifying social proximity.* Paper presented at the Sixth International Workshop on Peer-to-Peer Systems (IPTPS '07).
Freedman, M. J., Nissim, K., & Pinkas, B. (2004). *Efficient Private Matching and Set Intersection.* Paper presented at the Advances in Cryptology, Eurocrypt'04.
Gantz, J. F., Chute, C., Manfrediz, A., Minton, S., Reinsel, D., Schlichting, W., et al. (2008, March). The Diverse and Exploding Digital Universe: An Updated Forecast of Worldwide Information Growth Through 2011. from http://www.emc.com/collateral/analyst-reports/diverse-exploding-digital-universe.pdf
Gershenfeld, N., Krikorian, R., & Cohen, D. (2004). The Internet of Things. *Scientific American, 291*(4), 76-81.
GetSweet. GetSweet. Retrieved August 5, 2008, from http://www.getsweet.co.il/
Geurts, P., Khayat, I. E., & Leduc, G. (2004). *A Machine Learning Approach to Improve Congestion Control over Wireless Computer Networks.* Paper presented at the ICDM 2004.
Gips, J. P. (2006). *Social motion: Mobile networking through sensing human behavior.* Unpublished Master's thesis, Massachusetts Institute of Technology, Cambridge, MA.
Goffman, E. (1959). *The presentation of self in everyday life*. Garden City, N.Y.: Doubleday.
Goldberger, P. (2003). Disconnected Urbanism: The Cell Phone Has Changed Our Sense of Place More Than Faxes, Computers, and E-Mail. *Metropolis Magazine*.
Goldman, W. (1973). *The Princess Bride: S. Morgenstern's Classic Tale of True Love and High Adventure. the "Good Parts" Version, Abridged.*: Harcourt Children's Books.
Google. (2008). Google Talk for your iPhone device. Retrieved August 8, 2008, from http://www.google.com/mobile/apple/talk/
Google My Location. Retrieved July, 2008, from http://www.google.com/mobile/gmm/mylocation/index.html

Google Open Social. Retrieved August 5, 2008, from http://code.google.com/apis/opensocial/

Granovetter, M. (1973). The Strength of Weak Ties. *The American Journal of Sociology, 78*(6), 1360-1380.

Greenstein, B., McCoy, D., Pang, J., Kohno, T., Seshan, S., & Wetherall, D. (2008). *Improving wireless privacy with an identifier-free link layer protocol.* Paper presented at the 6th international conference on Mobile systems, applications, and services, Breckenridge, CO, USA.

Guttman, E., Perkins, C., Veizades, J., & Day, M. (1999). *Service Location Protocol, Version 2*: RFC Editor.

Hall, D. from http://animated-views.com/2007/disney-paris-exhibition/37-a-david-hall-watercolor-for-peter-pan-showing-his-sourcejpg/

Hall, P., & Pain, K. (2006). *Polycentric Metropolis: Learning from Mega-City Regions in Europe*. London: Earthscan.

Hampton, K. N. (2004). *Network sociability: on-line, off-line.* Northampton, MA: Edward Elgar.

Hampton, K. N., & Gupta, N. (In Press). Community and Social Interaction in the Wireless City: Wi-Fi use in Public and Semi-Public Spaces. *New Media & Society.*

Hancock, J. T., Thom-Santelli, J., & Ritchie, T. (2004). *Deception and design: the impact of communication technology on lying behavior.* Paper presented at the SIGCHI Conference on Human Factors in Computing Systems, Vienna, Austria.

Harris, K. (2003). Keep Your Distance: Remote Connection. *Journal of Community Work and Development,, 4.*

Harte, L. (2003). Introduction to Data Networks: PAN, LAN, WAN, and Wireless Data Technologies and System.

Hazay, C., & Lindell, Y. (2008). *Efficient Protocols for Set Intersection and Pattern Matching with Security Against Malicious and Covert Adversaries.* Paper presented at the Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC 2008, New York.

Heider, F. (1926). Thing and Medium. *Symposion, 1*, 109-157.

Hiertz, G. R. M., S., Zhao, R., Denteneer, D., & Berlemann, L. (2007). *Principles of IEEE 802.11s.* Paper presented at the 16th International Conference on Computer Communications and Networks, 2007. ICCCN 2007.

Hirsch, T., & Henry, J. (2005). *TXTmob: text messaging for protest swarms.* Paper presented at the CHI '05 extended abstracts on Human factors in computing systems.

Holmquist, L., Falk, J., & Wigstroem, J. (1999). Supporting Group Collaboration with Inter-Personal Awareness Devices. *Journal of Personal Technologies, 3*(1-2), 105-124.

ICQ. (2008). ICQ Wireless. Retrieved August 8, 2008, from http://download.icq.com/download/wireless/

IEEE 802.11. IEEE 802.11 LAN/MAN Wireless LANS. from http://standards.ieee.org/getieee802/802.11.html

Introducing JSON. Retrieved August 8, 2008, from http://www.json.org/

IrDA. IrDA Developers page. from http://www.irda.org/displaycommon.cfm?an=5

Jaiku. Retrieved July, 2008, from http://www.jaiku.com

Jones, S., & Kostakos, V. Cityware. from http://www.cityware.org.uk/

Kalofonos, D. N., Antoniou, Z., Reynolds, F. D., Van-Kleek, M., Strauss, J., & Wisner, P. (2008). *MyNet: A Platform for Secure P2P Personal and Social Networking Services.* Paper presented at the Sixth Annual IEEE International Conference on Pervasive Computing and Communications

Katti, S., Rahul, H., Hu, W., Katabi, D., Medard, M., & Crowcroft, J. (2006). *XORs In The Air: Practical Wireless Network Coding.* Paper presented at the ACM SIGCOMM 2006.

Kim, D., Peterson, N. M., Kim, J., Tabrizi, H., Burke, J. A., & Estrin, D. (2007). *Scalable System Design for Assisted Recall: Leveraging everyday mobile phones and web services*: Center for Embedded Network Sensing.

Kissner, L., & Song, D. X. (2005). *Privacy-Preserving Set Operations.* Paper presented at the Advances in Cryptology: Crypto.

Knight. Knight Foundation News Challenge. from http://www.newschallenge.org/

Koch, F. L., & Westphall, C. B. (2001). Decentralized Network Management Using Distributed Artificial Intelligence. *J. Network System Management, 9*(4), 375-388.

Kraut, R. E., Patterson, M., Lundmark, V., Kiesler, S., Mukhopadhyay, T., & Scherlis, W. (1998). Internet paradox: A social technology that reduces social involvement and psychological well-being? *American Psychologist, 53*(9), 1017-1032.

Krikorian, R., & Gershenfeld, N. (2004). Internet 0: Inter-Device Internetworking. *BT Technology Journal, 22*(4), 278-284.

Lai, K., Feldman, M., Stoica, I., & Chua, J. (2003). *Incentives for cooperation in peer-to-peer networks.* Paper presented at the Proceedings of the Workshop on Economics of Peer-to-Peer Systems, Berkeley, CA, USA.

Laibowitz, M., Gips, J., Aylward, R., Pentland, A., & Paradiso, J. A. (2006). *A sensor network for social dynamics*. Paper presented at the Proceedings of the fifth international conference on Information processing in sensor networks.

Lakoff, G. (1987). *Women, fire, and dangerous things: What categories reveal about the mind*. Chicago: University of Chicago Press.

Lam, B. (2007). LightBlue: a cross-platform Python Bluetooth API. from http://lightblue.sourceforge.net/

Lillie, A. S. (2007). Visualizing Music. Retrieved August 5, 2008, from http://flyingpudding.com/projects/viz_music/

Lillie, A. S. (2008). *MusicBox: Navigating the space of your music.* Unpublished Master's Thesis, Massachusetts Institute of Technology, Cambridge, MA.

Loopt. Loopt.com. Retrieved July 19, 2008, from http://loopt.com/

Ma, R. T. B., Lee, S. C. M., Lui, J. C. S., & Yau, D. K. Y. (2004). A game theoretic approach to provide incentive and service differentiation in P2P networks. *SIGMETRICS Performance Evaluation Review, 32*(1).

Madan, A. (2008). MobMedia. Retrieved August 5, 2008, from http://mob.media.mit.edu/info/index.php

Madan, A., & Pentland, A. (2006). *VibeFones: Socially Aware Mobile Phones.* Paper presented at the Wearable Computers, 2006 10th IEEE International Symposium on.

Madden, M., Fox, S., Smith, A., & Vitak, J. (2007). *Digital Footprints: Online identity management and search in the age of transparency*. Washington DC: PEW INTERNET & AMERICAN LIFE PROJECT.

MadWifi. Retrieved August 8, 2008, from http://madwifi.org/

Marti, S., Ganesan, P., & Garcia-Molina, H. (2004). *SPROUT: P2P routing with social networks* Paper presented at the International Workshop on Peer-to-Peer Computing & DataBases (P2P&DB 2004).

McLuhan, M. (1964). *Understanding Media: The Extensions of Man* (REV edition (Oct 20 1994) ed.): The MIT Press.

McMillan, R. (2008, July 17). Facebook bug leaks members' birthday data. *NetworkWorld*, from http://www.networkworld.com/news/2008/071608-facebook-bug-leaks-members-birthday.html

Meet Moi. www.meetmoi.com. Retrieved July, 2008, from http://www.meetmoi.com
Meyrowitz, J. (1985). *No sense of place : the impact of electronic media on social behavior.* New York: Oxford University Press.
Microsoft. (2007). Windows Live™ Messenger for mobile. from http://www.gowindowslive.com/Mobile/Landing/Messenger/Default.aspx?Locale=en-US
Miklas, A. G., Gollu, K. K., Chan, K. K. W., Saroiu, S., Gummadi, K. P., & Lara, E. d. (2007). *Exploiting Social Interactions in Mobile System.* Paper presented at the Proceedings of the Ninth International Conference on Ubiquitous Computing (Ubicomp), Innsbruck, Austria.
Milgram, S. (1970). The experience of living in cities. *Science*, 1461-1468.
Milgram, S. (1977). The Familiar Stranger: An Aspect of Urban Anonymity. In S. Milgram (Ed.), *The individual in a social world* (pp. 51-53). Reading, MA: Addison-Wesley.
Miller, J. H., & Page, S. E. (2007). *Complex Adaptive Systems*: Princeton University Press.
Minsky, M. (1988). *Society of Mind*: Simon & Schuster.
Minsky, M. (2006). *The Emotion Machine: Commonsense Thinking, Artificial Intelligence, and the Future of the Human Mind*: Simon & Schuster.
Mitchell, W. J. (1995). *City of bits: space, place, and the infobahn*: MIT Press.
Mitola, J. (2006). *Cognitive Radio Architecture: The Engineering Foundations of Radio XML*: Wiley.
Murph, D. (2006). Nike and Apple Launch Nike+iPod Sport Kit (for real). Retrieved August 8, 2008, from http://www.engadget.com/2006/07/13/nike-and-apple-launch-nike-ipod-sport-kit-for-real/
MySpace. from http://myspace.com
Nass, C., & Steuer, J. (1993). Voices, boxes, and sources of messages: Computers and social actors. *Human Communication Research, 19*(4), 504-527.
NFC Forum. NFC Forum homepage. from http://www.nfc-forum.org/home
Nidd, M. (2001). Service Discovery in DEAPspace. *IEEE Personal Communications*(August), 39-45.
Nie, N. H., & Erbring, L. (2002). Internet and Society: A Preliminary Report. *IT & Society, 1*(1), 275-283.
Nsyght. Retrieved 5 August, 2008, from http://nsyght.com/
NTag. from http://www.ntag.com/
Olguín, D. O. (2007). *Sociometric Badges: Wearable Technology for Measuring Human Behavior.* Unpublished Master's thesis, Massachusetts Institute of Technology, Cambridge, MA.
Olguin, D. O., & Pentland, A. (2007). *Sociometric Badges: State of the Art and Future Applications.* Paper presented at the IEEE 11th International Symposium on Wearable Computing, Boston, MA.
Oulasvirta, A., Tamminen, S., Roto, V., & Kuorelahti, J. (2005). *Interaction in 4-second bursts: The fragmented nature of attentional resources in mobile HCI.* Paper presented at the Conference on Human Factors in Computing Systems (CHI'05).
Oviatt, S. (2006). *Human-centered design meets cognitive load theory: designing interfaces that help people think.* Paper presented at the 14th annual ACM international conference on Multimedia, New York, NY, USA.
Pang, J., Greenstein, B., Gummad, R., Seshan, S., & Wetherall, D. (2007). *802.11 Client Identity Leakage*.

Pang, J., Greenstein, B., McCoy, D., Seshan, S., & Wetherall, D. (2007). *Tryst: The Case for Confidential Service Discovery*. Paper presented at the Sixth Workshop on Hot Topics in Networks (HotNets VI).

Paulos, E., & Goodman, E. (2004a, April). *The Familiar Stranger: Anxiety, Comfort, and Play in Public Places.* Paper presented at the ACM SIGCHI.

Paulos, E., & Goodman, E. (2004b). Jabberwocky. Retrieved August 8, 2008, from http://www.urban-atmospheres.net/Jabberwocky/demo.htm

Pentland, A., Gips, J., Dong, W., & Stoltzman, W. (2006). *Human computing for interactive digital media.* Paper presented at the Proceedings of the 14th annual ACM international conference on Multimedia, Santa Barbara, CA, USA.

Peter Pan and his shadow. Disney Auctions.

Putnam, R. D. (2000). *Bowling Alone: The Collapse and Revival of American Community*. New York: Simon and Schuster.

Quan Haase, A., Wellman, B., Hampton, K., & Witte, J. (2002). Internet , Social Capital, and Information Seeking. In B. Wellman & Haythorn-thwaite (Eds.), *The Internet in Everyday Life* (pp. 291-324). Oxford: C. Blackwell.

Raento, M., & Oulasvirta, A. (2005). *Privacy management for social awareness applications.* Paper presented at the Workshop on Context Awareness for Proactive Systems, CAPS.

Reed, D. P. (2001). The Law of the Pack. *Harvard Business Review*, 23-24.

Reed, D. P. (2006, unpublished). An Open Mobile Applications Platform for MIT's Living the Future (OMAP-LTF).

Rice, R. E. (2002). Primary issues in internet use: access, civic and community, involvement, and social interaction and expression. In L. A. Lievrouw & S. M. Livingstone (Eds.), *Handbook of New Media: Social Shaping and Consequences of ICTs* (pp. 105-129). London: Sage Publications.

Ricker, T. (2008). Best Buy Bluetooth Pairing Service Image. Retrieved August 8, 2008, from http://www.engadget.com/2008/07/14/best-buys-10-bluetooth-headset-pairing-service-includes-testin/

Rivest, R. L., Shamir, A., & Tauman, Y. (2001). *How to Leak a Secret.* Paper presented at the 7th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology.

Ruiz, P. M., Botia, J. A., & Gomez-Skarmeta, A. (2004). Providing QoS through machine-learning-driven adaptive multimedia applications. *IEEE Transactions on Systems, Man, and Cybernetics, Part B, 34*(3), 1398-1411.

Schneier, B. (1996). *Applied cryptography : protocols, algorithms, and source code in C* (2nd ed.). New York: Wiley.

SGT2.jpg. from http://www.hyperfunkalicious.net/ipb/index.php?showtopic=3693&st=20&p=48363&#entry48363

Tanenbaum, A. S. (2003). *Computer Networks* (Fourth Edition ed.). Saddle River, New Jersey: Prentice Hall.

TJX Says Theft of Credit Data Involved 45.7 Million Cards. (2007). *New York Times*, from TJX Says Theft of Credit Data Involved 45.7 Million Cards

TransferJet Consortium. (2008, July 17). Press release of the launch of TransferJet Consortium. from http://www.transferjet.org/en/news/press/200807_001.html

Tucker, E. (2008). Web networking photos come back to bite defendants. *Yahoo! News*, from http://news.yahoo.com/s/ap/20080720/ap_on_hi_te/tec_facebook_evidence

Twitter. Retrieved July, 2008, from http://www.twitter.com

Verizon Chaperone. Retrieved July, 2008, from http://www.verizonwireless.com/b2c/splash/chaperone/index.jsp

Viégas, F. B. (2000). *Collections : adapting the display of personal objects for different audiences.* Unpublished Thesis S.M. --Massachusetts Institute of Technology School of Architecture and Planning Program in Media Arts and Sciences 2000., Massachusetts Institute of Technology, Cambridge, MA.

Wellman, B. (1999). The Network Community. In B. Wellman (Ed.), *Networks in the Global Village* (pp. 1-48). Boulder, CO: Westview Press.

Wellman, B. (2001). Physical place and cyber-place: The rise of networked Individualism. *International Journal for Urban and Regional Research, 25*, 227-252.

Wicker, A. W. (1987). Behavior settings reconsidered: Temporal stages, resources, internal dynamics, context.x. In D. Stokols & I. Altman (Eds.), *Handbook of environmental psychology* (pp. 613-653). New York: John Wiley & Sons.

Williams, A. (2002). Zero Configuration Networking, Internet-Draft: Requirements for Automatic Configuration of IP Hosts. from http://files.zeroconf.org/draft-ietf-zeroconf-reqts-12.txt

WiMAX Forum. WiMAX Forum Technical Documents. from http://www.wimaxforum.org/technology/documents/

Wirth, L. (1938). Urbanism as a Way of Life. *The American Journal of Sociology, 44*(1), 1-24.

The Wizard of Oz movie. (1939).

Woyach, K., Puccinelli, D., & Haenggi, M. (2006). *Sensorless Sensing in Wireless Networks: Implementation and Measurements.* Paper presented at the Second International Workshop on Wireless Network Measurement (WinMee'06), Boston, MA.

Wright, H. F., & Barker, R. G. (1959). *Methods in Psychological Ecology*. Topeka, Kansas: Bay's Printing Service.

Yahoo Research. (2006). ZoneTag Retrieved July, 2008, from http://zonetag.research.yahoo.com/

Yahoo Research. (2008). Fire Eagle. Retrieved August 8, 2008, from http://fireeagle.yahoo.net/

Ypodimatopoulos, P. P., Reed, D. P., Lippman, A., & Bletsas, M. (2008). *Presence Information in Mobile Mesh Networks.* Paper presented at the 5th IEEE Consumer Communications and Networking Conference, 2008. CCNC 2008., Las Vevas, NV.

Zhu, F., Mutka, M. W., & Ni, L. M. (2005). Service discovery in pervasive computing environments. *Pervasive Computing, IEEE, 4*(4), 81- 90.

Zigbee Alliance. Zigbee Alliance homepage. from http://www.zigbee.org/en/index.asp

Zigelbaum, J., Kumpf, A., Alejandro, V., & Ishii, H. (2008). *Slurp: tangibility spatiality and an eyedropper.* Paper presented at the CHI '08 extended abstracts on Human factors in computing systems.

Zimmerman, H. (1980). OSI Reference Model — The ISO Model of Architecture for Open Systems Interconnection *IEEE Transactions on Communications, 28*(4), 425 – 432.

Zimmerman, T. G. (1996). Personal area networks: near-field intrabody communication. *IBM Syst. J., 35*(3-4), 609-617.

Zimmermann, P. (1995). *PGP Source Code and Internals*: MIT Press.

*"Well I'm on my way.*
*I don't know where I'm going.*
*I'm on my way,*
*I'm taking my time but I don't know where."*

*-Paul Simon, Me and Julio Down by the Schoolyard (1971)*